ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG

NGUYỄN NGỌC NHUNG
HÀ THỊ THU HIỀN
PHẠM NGỌC THƠ

ĐỒ ÁN MÔN PHÁP CHỨNG KỸ THUẬT SỐ

THU THẬP DỮ LIỆU ỨNG DỤNG THREADS

Crawling Threads App Data

CỬ NHÂN NGÀNH AN TOÀN THÔNG TIN

TP. HỒ CHÍ MINH, 2024

ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG

NGUYỄN NGỌC NHUNG – 21521248
HÀ THỊ THU HIỀN – 21522056
PHẠM NGỌC THƠ - 21522641

ĐỒ ÁN MÔN PHÁP CHỨNG KỸ THUẬT SỐ

THU THẬP DỮ LIỆU ỨNG DỤNG THREADS

Crawling Threads App Data

CỬ NHÂN NGÀNH AN TOÀN THÔNG TIN

GIẢNG VIÊN HƯỚNG DẪN
NGHI HOÀNG KHOA
NGUYỄN TẤN CẦM

TP. HỒ CHÍ MINH, 2024

LỜI CẢM ƠN

Kính gửi Thầy Nghi Hoàng Khoa và Thầy Nguyễn Tấn Cầm,

Chúng em xin gửi lời cảm ơn chân thành nhất đến Thầy về sự tận tâm và hướng dẫn chúng em nhiệt tình trong môn học Pháp chứng Kỹ thuật số. Nhờ sự hỗ trợ của Thầy, chúng em đã không chỉ nắm vững lý thuyết về các công cụ và kỹ thuật điều tra số mà còn biết cách áp dụng chúng vào quá trình thực hiện đồ án. Qua đó, chúng em cũng đã hiểu rõ hơn về quy trình phân tích dữ liệu số và ý nghĩa của việc bảo vệ bằng chứng số trong công tác điều tra.

Sự tận tụy và sự hỗ trợ của Thầy đã tạo nên một môi trường học tập đầy cống hiến. Chúng em cũng xin bày tỏ lòng biết ơn đối với sự động viên và những lời khuyên quý giá từ Thầy trong quá trình làm đồ án. Thầy đã luôn sẵn lòng lắng nghe và giúp đỡ chúng em vượt qua những khó khăn.

Chúng em xin cam đoan sẽ tiếp tục áp dụng những kiến thức đã học từ Thầy vào công việc và cuộc sống hàng ngày. Sự hướng dẫn và đóng góp của Thầy sẽ mãi là nguồn động lực và định hướng cho chúng em trong tương lai.

Trân trọng,
Nguyễn Ngọc Nhung
Hà Thị Thu Hiền
Phạm Ngọc Thơ

MỤC LỤC

DANH MỤC HÌNH

Chương 1. TỔNG QUAN

1.1. Giới thiệu về Threads

Threads là một nền tảng mạng xã hội do Meta phát triển, được thiết kế để chia sẻ nội dung ngắn dạng văn bản, liên kết, video và hình ảnh. Mục đích chính của Threads là cung cấp một nền tảng mạng xã hội tập trung vào quyền riêng tư, cho phép người dùng chia sẻ nội dung và giao tiếp trong một môi trường được bảo mật.

1.2. Tầm quan trọng của việc thu thập dữ liệu ứng dụng Threads

Trong thời đại kỹ thuật số, việc sử dụng dữ liệu từ các nền tảng truyền thông xã hội là một phần không thể thiếu đối với các doanh nghiệp, nhà nghiên cứu và nhà tiếp thị. Với sức mạnh của cộng đồng người dùng và sự đa dạng của nội dung, Threads đã trở thành một nguồn thông tin quan trọng. Tuy nhiên, việc truy cập và thu thập dữ liệu từ Threads có thể gặp phải nhiều thách thức do tính phức tạp của cấu trúc và các biện pháp bảo mật của nền tảng. Để giải quyết vấn đề này, chúng ta sẽ tiến hành thu thập dữ liệu cùng với sự hỗ trợ của Python.

Dữ liệu được thu thập từ Threads có thể được sử dụng trong các mục đích:

- Phát hiện gian lận (Fraud Detection): Nhận diện các hành vi gian lận như mạo danh, tin tức giả, và hoạt động bất hợp pháp.

- Bảo mật mạng xã hội (Social Media Security): Giám sát các mối đe dọa và lỗ hổng bảo mật trên nền tảng.

- Phân tích tấn công mạng (Cyber Attack Analysis): Phân tích các kỹ thuật tấn công mạng của hacker để cải thiện biện pháp phòng ngừa.

- Quản lý rủi ro bảo mật (Security Risk Management): Đánh giá và quản lý rủi ro trong các chiến dịch truyền thông xã hội.

- Phân tích thông tin tình báo (Threat Intelligence Analysis): Xây dựng cơ sở dữ liệu về mối đe dọa bảo mật để phòng ngừa các cuộc tấn công.

- Giám sát hoạt động tội phạm mạng (Cybercrime Activity Monitoring): Hỗ trợ điều tra các hoạt động tội phạm mạng.

- Bảo vệ thương hiệu (Brand Protection): Giám sát và ngăn chặn các mối đe dọa đối với danh tiếng thương hiệu.

- Phân tích hành vi người dùng (User Behavior Analysis): Xác định hành vi bất thường để phát hiện sớm mối đe dọa.

- Kiểm soát truy cập (Access Control): Thiết lập và duy trì các chính sách kiểm soát truy cập.

- Phân tích lỗ hổng bảo mật (Vulnerability Analysis): Phát hiện và vá các lỗ hổng bảo mật trên nền tảng.

1.3. Mục đích nghiên cứu

Mục đích của nghiên cứu là tìm hiểu và áp dụng các kỹ thuật thu thập dữ liệu từ nền tảng Threads bằng Python nhằm phục vụ cho các phân tích trong lĩnh vực pháp chứng kỹ thuật số. Nghiên cứu nhằm xây dựng quy trình và công cụ thu thập dữ liệu hiệu quả, từ đó đảm bảo tính chính xác và bảo mật của dữ liệu thu thập được.

1.4. Đối tượng nghiên cứu

Đối tượng nghiên cứu của đề tài là các dữ liệu được chia sẻ và tương tác trên nền tảng Threads, bao gồm các bài viết, bình luận, lượt thích, chia sẻ và các hoạt động khác của người dùng. Nghiên cứu tập trung vào việc áp dụng Python để thu thập và phân tích các dữ liệu này nhằm phục vụ cho các mục đích pháp chứng kỹ thuật số.

1.5. Phạm vi nghiên cứu

Phạm vi nghiên cứu bao gồm các khía cạnh liên quan đến việc thu thập và phân tích dữ liệu trên nền tảng Threads, bao gồm các công nghệ và thuật toán sử dụng trong quá trình thu thập dữ liệu, quản lý và lưu trữ dữ liệu, và phân tích dữ liệu để rút ra các thông tin có giá trị. Nghiên cứu sẽ tập trung vào các phương pháp và công cụ hiện có để thu thập dữ liệu từ Threads bằng Python, đồng thời đề xuất cách giải quyết các thách thức và vấn đề có thể phát sinh trong quá trình triển khai thu thập dữ liệu.

Chương 2. PHÂN TÍCH HỆ THỐNG

2.1. Cấu trúc của một trang cá nhân Threads

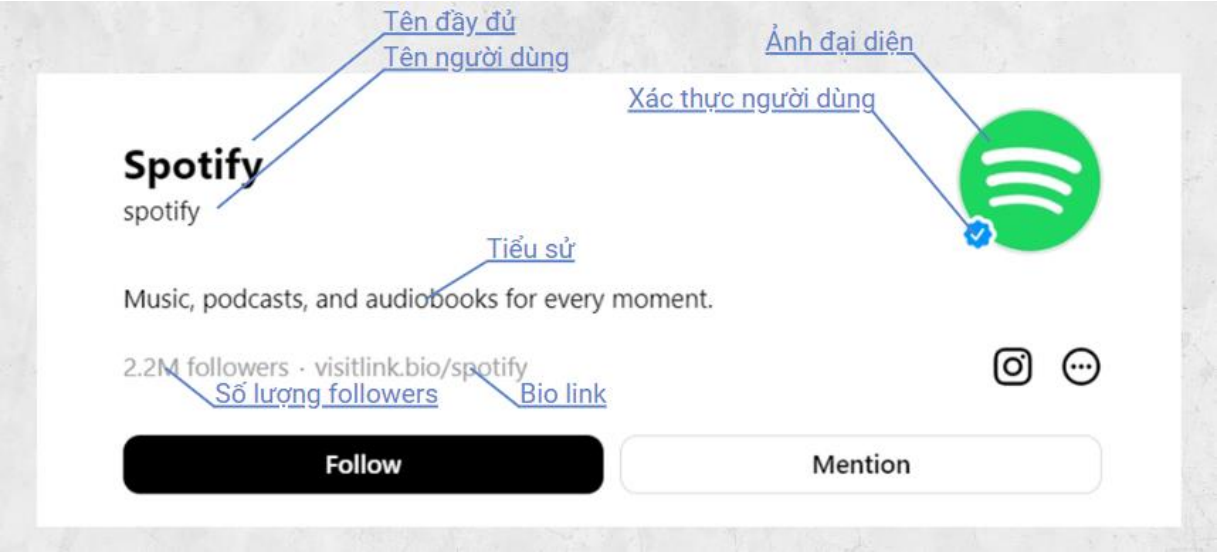

Figure 2.1. Ví dụ cho một trang cá nhân Threads

- Ảnh đại diện: Thường là một hình ảnh tròn nhỏ tượng trưng cho chủ tài khoản. Nhấp vào ảnh hồ sơ sẽ mở ra phiên bản lớn hơn của ảnh. Dấu tick xanh bên cạnh ảnh đại diện thể hiện tài khoản đã được xác thực.

- Tên đầy đủ và tên người dùng: Tên người dùng là mã định danh duy nhất cho tài khoản, thường viết không dấu và không có khoảng trắng. Tên đầy đủ là họ tên của người dùng, có thể viết có dấu và có khoảng trắng.

- Tiểu sử: Một mô tả ngắn mà người dùng có thể tùy chỉnh để cung cấp thông tin về bản thân, sở thích hoặc doanh nghiệp của họ.

- Bio link: Có thể đặt liên kết đến trang web cá nhân, blog, cửa hàng trực tuyến, hoặc các mạng xã hội khác ở đây. Đây là một cách hiệu quả để chia sẻ thêm thông tin hoặc hướng người xem đến các nội dung khác mà họ quan tâm.

- Số lượng followers: Số lượng người theo dõi, đo lường và đánh giá sức ảnh hưởng của một người dùng hoặc một tài khoản trên nền tảng.

2.2. Cấu trúc của một trang bài viết Threads

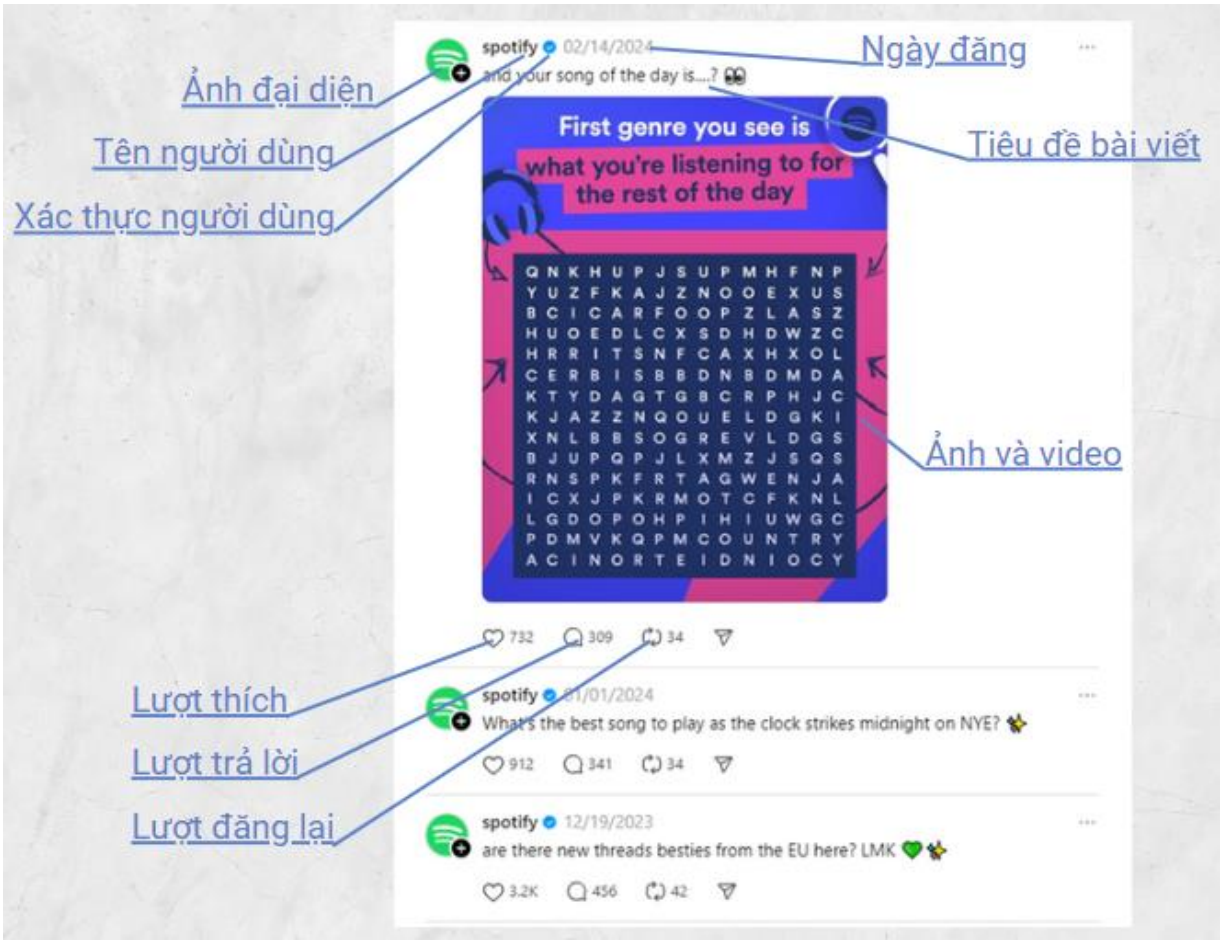

Figure 2.2. Ví dụ cho một trang bài viết Threads

- Tiêu đề bài viết: Mô tả ngắn gọn nội dung chính hoặc mục tiêu của bài viết.

- Ngày đăng: Ngày mà bài viết được đăng lên Threads, thường được hiển thị bên trên nội dung.

- Ảnh đại diện, tên người dùng và xác thực người dùng đã nêu ở trên.

- Ảnh và video: Các tệp đa phương tiện được chia sẻ, thường được sử dụng để truyền đạt thông điệp hoặc chia sẻ trải nghiệm với người khác.

- Lượt thích, lượt trả lời, lượt đăng lại: Các thao tác tương tác mà người dùng có thể thực hiện đối với một bài viết hoặc bài đăng trên mạng xã hội. Các con số này thường được sử dụng để đo lường mức độ tương tác và sức ảnh hưởng của nội dung.

2.3. Vấn đề pháp lý

Vấn đề pháp lý khi thu thập dữ liệu trên Threads là một phần quan trọng cần xem xét. Dữ liệu trên Threads được công khai và việc thu thập chúng là hợp pháp, nhưng điều này chỉ áp dụng khi việc thu thập được thực hiện ở mức độ hợp lý và không gây ra thiệt hại cho ứng dụng. Để đảm bảo tuân thủ các nguyên tắc, cần kiểm tra tệp robots.txt của Threads để biết các quy định về thu thập dữ liệu. Việc tuân thủ các quy định này không chỉ tránh được các vấn đề pháp lý mà còn đảm bảo tính đạo đức của hoạt động thu thập dữ liệu.

```
# Notice: Collection of data on Threads through automated means is
# prohibited unless you have express written permission from Threads
# and may only be conducted for the limited purpose contained in said
# permission.
User-agent: Applebot
Disallow: /accounts/
Disallow: /activity/
Disallow: /ajax/
Disallow: /client_error/
Disallow: /deck/
Disallow: /intent/
Disallow: /logging/
Disallow: /onboarding/
Disallow: /publicapi/
Disallow: /qp/batch_fetch_web/
Disallow: /query/
Disallow: /search/
Disallow: /settings/

User-agent: baiduspider
Disallow: /accounts/
Disallow: /activity/
Disallow: /ajax/
Disallow: /client_error/
Disallow: /deck/
Disallow: /intent/
Disallow: /logging/
Disallow: /onboarding/
Disallow: /publicapi/
Disallow: /qp/batch_fetch_web/
Disallow: /query/
Disallow: /search/
Disallow: /settings/
```

Figure 2.3. Nội dung file robots.txt

Chương 3. GIẢI PHÁP

3.1. Cài đặt thư viện

- Playwright là một thư viện kiểm thử đầu cuối (end-to-end testing) cho phép tự động hóa tương tác với trình duyệt web. Nó cho phép viết các kịch bản để điều khiển trình duyệt, như là điều hướng trang, nhấp chuột, và trích xuất thông tin từ trang web.

- Scrapfly SDK cung cấp các công cụ và API để hỗ trợ việc thu thập dữ liệu web (web scraping) một cách hiệu quả, bao gồm cả xử lý chống lại các biện pháp chống trích xuất của trang web.

- Jmespath là một thư viện để truy vấn và chuyển đổi dữ liệu JSON. Nó cho phép tìm kiếm, lọc và lấy dữ liệu cụ thể từ các đối tượng JSON phức tạp.

- Nested_lookup là một thư viện cho phép tìm kiếm và trích xuất các giá trị từ các cấu trúc dữ liệu lồng nhau (nested data structures) như JSON.

Cài đặt các thư viện bằng pip:

$ pip install playwright nested_lookup jmespath "scrapfly-sdk[all]"

3.2. Cách Threads hiển thị dữ liệu

Threads sử dụng dữ liệu web ẩn để tải thông tin bài viết. Nói cách khác, nó ẩn dữ liệu trong một phần tử <script> dưới dạng JSON và khi trang tải xong, nó sẽ mở rộng dữ liệu này thành phần HTML hiển thị của trang.

Để đảo ngược kỹ thuật này, chúng ta có thể sử dụng Công cụ Phát triển Trình duyệt (Browser Developer Tools), cho phép chúng ta kiểm tra toàn bộ trang web.

Figure 3.1. Source code của một trang cá nhân Threads chứa nhiều phần tử <script>

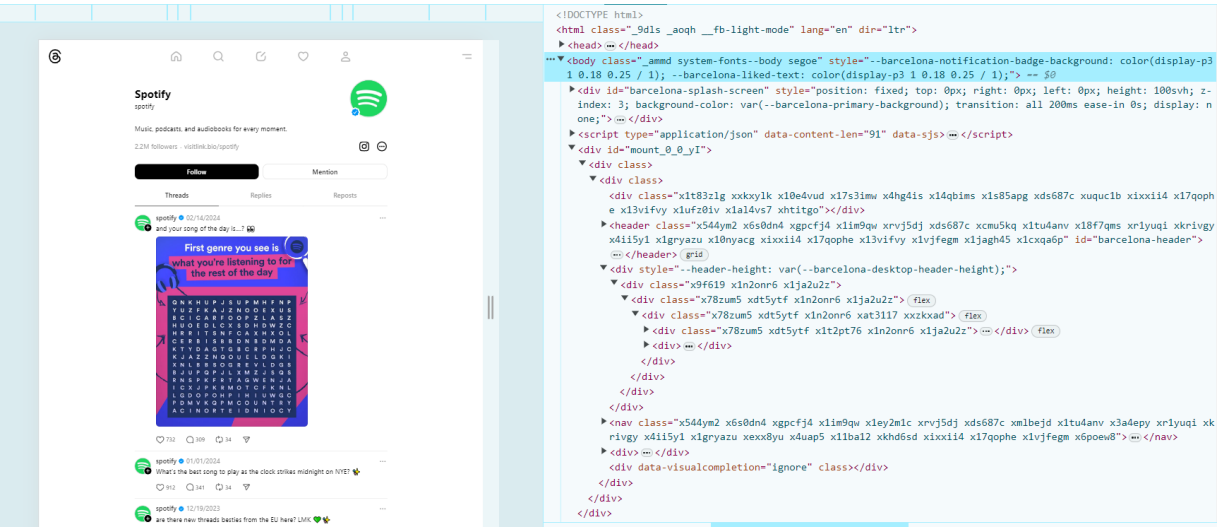

Figure 3.2. Khi trang tải xong, trang web hiển thị các thông tin ở dạng các phần tử html

Lấy ví dụ một phần văn bản được hiển thị trên trang là “and your song of the day is....?”.

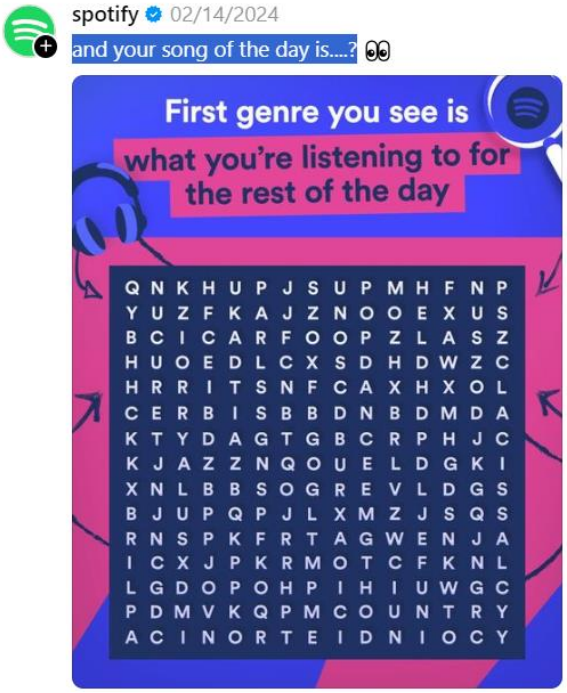

Figure 3.3. Phần văn bản được hiển thị trên trang

Tiến hành tìm kiếm văn bản này trong source code. Chúng ta có thể thấy văn bản được ẩn trong một phần tử <script> dưới dạng JSON.

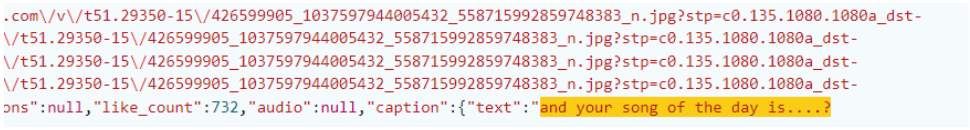

Figure 3.4. Phần văn bản được hiển thị trên source code

3.3. Thu thập dữ liệu trang cá nhân

Bước 1: Sử dụng Playwright để tải trang web một cách tự động mà không cần giao diện người dùng.

```
with sync_playwright() as pw:
    browser = pw.chromium.launch()
    context = browser.new_context(viewport={"width": 1920, "height": 1080})
    page = context.new_page()

    page.goto(url)
    page.wait_for_selector("[data-pressable-container=true]")
    selector = Selector(page.content())
```

Figure 3.5. Tải trang web không cần giao diện

Đầu tiên, Playwright được khởi động bằng cách sử dụng **sync_playwright()**, sau đó trình duyệt Chromium được mở thông qua lệnh **pw.chromium.launch()**. Tiếp theo, một ngữ cảnh trình duyệt mới với kích thước cửa sổ 1920x1080 được tạo ra bằng lệnh **browser.new_context(viewport={"width": 1920, "height": 1080})**. Sau đó, một tab trình duyệt mới được mở bằng lệnh **context.new_page()**, và trình duyệt điều hướng đến URL được cung cấp thông qua **page.goto(url)**. Trình duyệt sẽ chờ cho đến khi phần tử có selector **[data-pressable-container=true]** xuất hiện để đảm bảo rằng trang đã tải xong, sử dụng lệnh **page.wait_for_selector("[data-pressable-container=true]")**. Cuối cùng, nội dung HTML của trang được lấy và một đối tượng **Selector** được tạo ra để phân tích HTML đó, thông qua lệnh **Selector(page.content())**.

Bước 2: Sau khi tải trang, chúng ta sẽ sử dụng parsel để phân tích cấu trúc HTML của trang web và tìm kiếm phần tử <script>.

```
hidden_datasets = selector.css('script[type="application/json"][data-sjs]::text').getall()
```

Figure 3.6. Tìm các phần tử <script> phù hợp

Dòng mã này sử dụng đối tượng **Selector** để tìm và trích xuất tất cả các tập dữ liệu JSON ẩn trong các phần tử <script> có thuộc tính **type="application/json"** và **data-sjs** trên trang web. Kết quả là biến hidden_datasets chứa các tập dữ liệu JSON ẩn, sẵn sàng để được giải mã và phân tích nhằm thu thập thông tin cần thiết từ trang web.

Bước 3: Dựa trên nội dung của các phần tử <script>, chúng ta sẽ xác định phần tử nào chứa dữ liệu JSON cần tìm.

```
for hidden_dataset in hidden_datasets:
    if '"ScheduledServerJS"' not in hidden_dataset:
        continue
    is_profile = 'follower_count' in hidden_dataset
    is_threads = 'thread_items' in hidden_dataset
    if not is_profile and not is_threads:
        continue
    data = json.loads(hidden_dataset)
```

Figure 3.7. Xác định phần tử chứa dữ liệu JSON phù hợp

Đầu tiên, chương trình kiểm tra xem chuỗi JSON có chứa từ khóa **"ScheduledServerJS"** không. "ScheduledServerJS" là một dấu hiệu đặc trưng cho thấy chuỗi JSON bao gồm thông tin quan trọng mà chương trình đang tìm kiếm, như dữ liệu hồ sơ người dùng hoặc bài viết. Sau đó, kiểm tra xem chuỗi JSON có chứa thông tin hồ sơ người dùng (**'follower_count'**) hoặc thông tin bài viết (**'thread_items'**) không. Nếu không có cả hai, chuỗi sẽ bị bỏ qua. Chuỗi JSON chứa thông tin hồ sơ sẽ được giải mã bằng **json.loads()**.

Bước 4: Sau khi tìm thấy dữ liệu JSON, chúng ta sẽ sử dụng các thư viện như nested_lookup và jmespath để giải mã và trích xuất các thông tin cần thiết từ tập dữ liệu này.

```
if is_profile:
    user_data = nested_lookup('user', data)
    parsed['user'] = parse_profile(user_data[0])
```

Figure 3.8. Trích xuất dữ liệu người dùng

Dữ liệu người dùng sẽ được trích xuất bằng **nested_lookup('user', data)**. Sau đó, dữ liệu người dùng được phân tích và lưu vào biến **parsed['user']** thông qua hàm **parse_profile(user_data[0])**.

```python
def parse_profile(data: Dict) -> Dict:
    result = jmespath.search(
        """{
            is_private: text_post_app_is_private,
            is_verified: is_verified,
            profile_pic: hd_profile_pic_versions[-1].url,
            username: username,
            full_name: full_name,
            bio: biography,
            bio_links: bio_links[].url,
            followers: follower_count
        }""",
        data,
    )
    result["url"] = f"https://www.threads.net/@{result['username']}"
    return result
```

Figure 3.9. Phân tích dữ liệu theo các trường

Hàm **parse_profile** phân tích dữ liệu hồ sơ người dùng từ một từ điển JSON. Nó sử dụng **jmespath** để trích xuất các trường quan trọng như trạng thái riêng tư, xác thực, ảnh hồ sơ, tên người dùng, tên đầy đủ, tiểu sử, liên kết trong tiểu sử, và số lượng người theo dõi. Sau đó, hàm tạo một URL hồ sơ từ tên người dùng và trả về kết quả dưới dạng từ điển.


```
  "user": {
    "is_private": false,
    "is_verified": true,
    "profile_pic": "https://scontent.cdninstagram.com/v/t51.2885-19/358162502_10
38284560473867_7645518712425998110_n.jpg?stp=dst-jpg_s320x320&_nc_ht=scontent.cd
ninstagram.com&_nc_cat=103&_nc_ohc=fWXpPryYuZAQ7kNvgFTPM1r&edm=APs17CUBAAAA&ccb=
7-5&oh=00_AYA0wlTA_K3mVxebDNh1MUK8SOOQSkW14plSooO1VWM5hA&oe=66554302&_nc_sid=10d
13b",
    "username": "spotify",
    "full_name": "Spotify",
    "bio": "Music, podcasts, and audiobooks for every moment.",
    "bio_links": [
      "http://visitlink.bio/spotify/"
    ],
    "followers": 2283087,
    "url": "https://www.threads.net/@spotify"
  },
```


Figure 3.10. Kết quả thu thập dữ liệu trang cá nhân

3.4. Thu thập dữ liệu bài viết

Các bước thực hiện thu thập dữ liệu bài viết tương tự như thu thập dữ liệu trang cá nhân.

Bước 1: Sử dụng Playwright để tải trang web một cách tự động mà không cần giao diện người dùng.

Bước 2: Sau khi tải trang, chúng ta sẽ sử dụng parsel để phân tích cấu trúc HTML của trang web và tìm kiếm phần tử <script>.

Bước 3: Dựa trên nội dung của các phần tử <script>, chúng ta sẽ xác định phần tử nào chứa dữ liệu JSON cần tìm.

Bước 4: Sau khi tìm thấy dữ liệu JSON, chúng ta sẽ sử dụng các thư viện như nested_lookup và jmespath để giải mã và trích xuất các thông tin cần thiết từ tập dữ liệu này.

```
if is_threads:
    thread_items = nested_lookup('thread_items', data)
    threads = [
        parse_thread(t) for thread in thread_items for t in thread
    ]
    parsed['threads'].extend(threads)
```

Figure 3.11. Trích xuất dữ liệu bài viết

Sử dụng hàm **nested_lookup** để tìm kiếm và trích xuất danh sách các mục **thread_items** từ dữ liệu JSON. Với mỗi mục trong danh sách thread_items, chương trình sẽ thực hiện phân tích chi tiết bằng cách gọi hàm **parse_thread**. Kết quả cuối cùng là một danh sách các bài viết đã được phân tích được gộp vào biến **parsed['threads']**.

```
def parse_thread(data: Dict) -> Dict:
    result = jmespath.search(
        """{
            text: post.caption.text,
            published_on: post.taken_at,
            id: post.id,
            pk: post.pk,
            code: post.code,
            username: post.user.username,
            user_pic: post.user.profile_pic_url,
            user_verified: post.user.is_verified,
            user_pk: post.user.pk,
            user_id: post.user.id,
            has_audio: post.has_audio,
            like_count: post.like_count,
            reply_count: post.text_post_app_info.direct_reply_count,
            images: post.carousel_media[].image_versions2.candidates[1].url || post.image_versions2.candidates[1].url,
            videos: post.video_versions[].url
        }""",
        data,
    )
    result["videos"] = list(set(result["videos"] or []))
    if result["reply_count"] and type(result["reply_count"]) != int:
        result["reply_count"] = int(result["reply_count"].split(" ")[0])
    result[
        "url"
    ] = f"https://www.threads.net/@{result['username']}/post/{result['code']}"
    return result
```

Figure 3.12. Phân tích dữ liệu theo các trường

Trong hàm **parse_thread**, chúng ta sử dụng JMESPath để trích xuất thông tin từ dữ liệu đầu vào. Truy vấn này giúp chúng ta lấy các chi tiết quan trọng như văn bản, thời gian xuất bản, ID, và các đường dẫn đến hình ảnh và video. Sau đó, chúng ta điều chỉnh dữ liệu như loại bỏ video trùng lặp và chuyển **reply_count** thành số nguyên nếu cần thiết. Cuối cùng, chúng ta tạo một URL đến bài viết trên Threads.net để dễ dàng truy cập.

```
  "threads": [
    {
      "text": "and your song of the day is....? 👀",
      "published_on": 1707850800,
      "id": "3302014303841616046_224223453",
      "pk": "3302014303841616046",
      "code": "C3THIO3vQCu",
      "username": "spotify",
      "user_pic": "https://scontent.cdninstagram.com/v/t51.2885-19/358162502_1038284560473867_7645518712425998110_n.
jpg?stp=dst-jpg_s150x150&_nc_ht=scontent.cdninstagram.com&_nc_cat=103&_nc_ohc=fWXpPryYuZAQ7kNvgEoFrqj&edm=APs17CUBAA
AA&ccb=7-5&oh=00_AYDCrHjPYzq_Akht70S7INCgXVQjWjBKkS1_XtlxTh7Fpw&oe=66557B42&_nc_sid=10d13b",
      "user_verified": true,
      "user_pk": "224223453",
      "user_id": null,
      "has_audio": null,
      "like_count": 732,
      "reply_count": 309,
      "images": "https://scontent.cdninstagram.com/v/t51.29350-15/426599905_1037597944005432_558715992859748383_n.jp
g?stp=dst-jpg_e35_p720x720&efg=eyJ2ZW5jb2RlX3RhZyI6ImltYWdlX3VybGdlbi4xMDgweDEzNTAuc2RyLmYyOTM1MCJ9&_nc_ht=scontent.
cdninstagram.com&_nc_cat=100&_nc_ohc=1RnP1hCeK44Q7kNvgGSy2gg&edm=APs17CUBAAAA&ccb=7-5&ig_cache_key=MzMwMjAxNDMwMzg0M
TYxNjA0Ng%3D%3D.2-ccb7-5&oh=00_AYCpbAe-Bjk_x-vtC_EF4e_bi-WVtbYG20fl1s0FGEf8vA&oe=6653828C&_nc_sid=10d13b",
      "videos": [],
      "url": "https://www.threads.net/@spotify/post/C3THIO3vQCu"
    },
    {
      "text": "What's the best song to play as the clock strikes midnight on NYE? ✨",
      "published_on": 1704043272,
      "id": "3270074442556888690_224223453",
      "pk": "3270074442556888690",
      "code": "C1ho2FBr9Zy",
      "username": "spotify",
      "user_pic": "https://scontent.cdninstagram.com/v/t51.2885-19/358162502_1038284560473867_7645518712425998110_n.
jpg?stp=dst-jpg_s150x150&_nc_ht=scontent.cdninstagram.com&_nc_cat=103&_nc_ohc=fWXpPryYuZAQ7kNvgEoFrqj&edm=APs17CUBAA
AA&ccb=7-5&oh=00_AYDCrHjPYzq_Akht70S7INCgXVQjWjBKkS1_XtlxTh7Fpw&oe=66557B42&_nc_sid=10d13b",
      "user_verified": true,
      "user_pk": "224223453",
      "user_id": null,
      "has_audio": null,
      "like_count": 912,
      "reply_count": 341,
      "images": null,
      "videos": [],
      "url": "https://www.threads.net/@spotify/post/C1ho2FBr9Zy"
    },
```


Figure 3.13. Kết quả thu thập dữ liệu bài viết

Trong kết quả trả về, "published_on": 1707850800 là một giá trị đại diện cho thời gian được biểu diễn dưới dạng Unix timestamp. Unix timestamp là một cách để đại diện cho thời gian như một số nguyên, bắt đầu từ 00:00:00 UTC ngày 1 tháng 1 năm 1970.