Web Scraping using BeautifulSoup, Selenium, and Wireshark

2022023 Dongmyeong Back

1. Website Selection

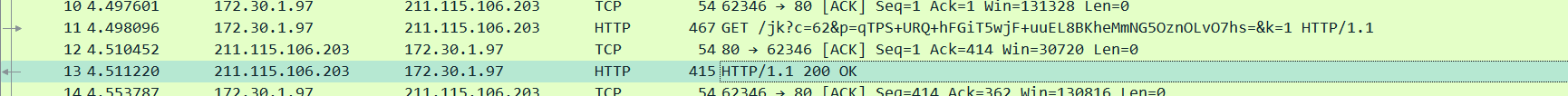
For this project, I selected an Instagram hashtag page focused on #홍대맛집 (Hongdae restaurants) as the target website. The goal was to scrape results for Hongdae restaurant recommendations. Given that the page is dynamic and interactive, I employed Selenium for thorough analysis. This allowed me to navigate and extract data effectively from the dynamic content of the Instagram hashtag page, ensuring accurate data retrieval for further analysis.

websiteURL: [www.instagram.com](http://www.instagram.com)

1. Packet Capture using Wireshark

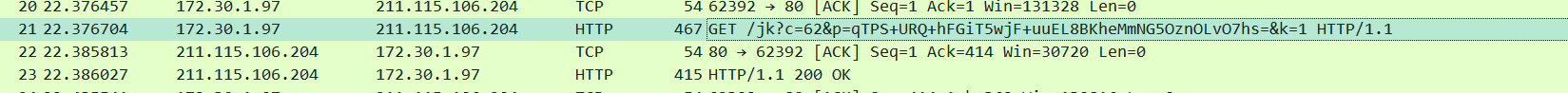
Wireshark captures were performed in a Wi-Fi environment while logging into Instagram and navigating to the hashtag page. Specifically, I focused on capturing TCP and HTTP packets at the moment when the transition to the hashtag page occurred. Analyzing multiple packet sequences will allow me to infer the content exchanged during these critical interactions.

* Go to [www.instagram.com](http://www.instagram.com)



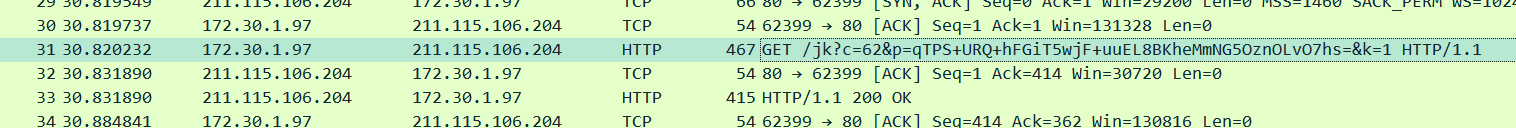
I was able to identify that my computer's IP address is 172.30.1.97, while the Instagram server's IP address is 211.115.106.203. During the process of loading the www.instagram.com page, I observed that an HTTP GET request was sent at line 11. Subsequently, at line 13, an HTTP response with a successful 200 status code indicated that the request was well-received. It's also worth noting that before the HTTP communication, there were multiple TCP communications established with the same address, suggesting the establishment of connections prior to the actual HTTP request.

* Log in



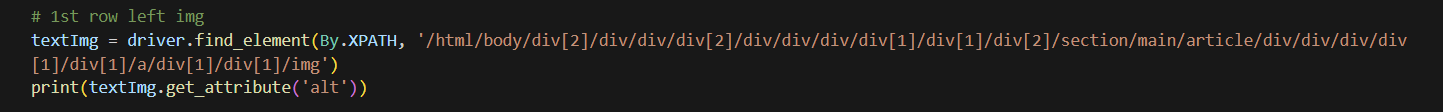
At line 21, I observed an HTTP communication occurring right after clicking the login button. This time, a GET request was sent to the IP address 211.115.106.204. Similarly, at line 23, a 200 status code in the HTTP response indicated a successful login process. This sequence of events confirms that the login operation was accomplished effectively.

* Go to hastag page



Line 31 represents an HTTP communication that requested the hashtag page. A GET request was sent to the IP address 211.115.106.204, and it's evident that the hashtag page was successfully loaded. This demonstrates the effective retrieval of the hashtag page through the HTTP request.

1. HTML Structure Analysis



To extract the information I was interested in, which is the names of restaurants, I needed to locate relevant Instagram posts. Upon analyzing the HTML structure, I discovered that the information I needed was also embedded within the `img` tags. Consequently, I copied the XPath of the `img` tag and used Selenium to retrieve it. By using the `get\_attribute` method, I accessed the 'alt' attribute, which serves as an alternative text when the image cannot be displayed. This approach allowed me to successfully gather the restaurant names from the 'alt' attribute of the `img` tags in the Instagram posts.

스크린샷, 텍스트, 폰트이(가) 표시된 사진

자동 생성된 설명

Recognizing that not all 'img' tags had meaningful 'alt' attributes, which are not always present unless explicitly set by users, I sought a comprehensive approach to gather data. To achieve this, I also aimed to extract content from the Instagram posts themselves. This required a distinct method, as it involved the process of clicking on each individual post. To do this, I obtained the XPath for clicking on posts and then extracted text content from the 'h1' class element. This dual approach ensured a more extensive data collection, encompassing both images with 'alt' attributes and textual information from the posts.

1. Data Retrieval

**The ‘alt’ attribute of img tag**

In this assignment, the data I aimed to collect was the names of restaurants that appear in the results of hashtag searches. To simplify this task, I initially focused on locating the 'alt' attribute within the 'img' tags. This attribute serves as a substitute text when images cannot be displayed. The advantage was that I could extract this information directly from the HTML of the hashtag page without the need to manually click on each individual post.

텍스트, 폰트, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

Each HTML element had a structured XPath address, and by applying a repetitive process, I systematically retrieved all the relevant data. To store the gathered information in a CSV format, I organized the data into an array, including the collection date as part of each record.

텍스트, 멀티미디어 소프트웨어, 소프트웨어, 스크린샷이(가) 표시된 사진

자동 생성된 설명

**The Instagram id and contents of the post**

Since not everyone provides meaningful 'alt' attributes for images, I opted to click on individual posts to retrieve their content. Additionally, to prevent duplicating the analysis of the same posts later on, I also collected the Instagram IDs of the users who uploaded the posts. This approach ensured a more comprehensive data collection strategy.

텍스트, 소프트웨어, 폰트, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

Similarly, upon analyzing the XPath structure within the HTML, I identified a pattern that allowed for systematic processing using loops. However, I encountered issues with the `find\_element().click()` method within the loop. As a solution, I opted to locate the element with `find\_element()` and then used `execute\_script` to simulate the click action. The data collected through this method was then appended to the previously created array.

텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

1. Data Storage

All the data obtained through web scraping has been meticulously organized and stored in an array called "todayDatas." Within this array, individual records consist of date, 'alt' text, user IDs, and content. To ensure data continuity over several days, I've employed CSV files, adding column names, and saving the data into a file named "data.csv." This dataset will prove invaluable for our upcoming team project, which involves collecting and analyzing Instagram posts related to restaurant recommendations. Our objective is to extract restaurant names from these posts and conduct in-depth analysis using this rich dataset.