Website Scraper

INSTRUCTOR: DR. RABIA LATIF

COURSE: CYS401

NAMES: HAIFA ZEIN EDDIN, MARYAH ALREBDI, GHADA ALBUGAMI

Contents

Introduction

Website scraping refers to the process of automatically collecting wide ranging amounts of online data from related websites in a short amount of time using scraping software. The ability to scrape or collect large amounts of data is done by two continuous fragments, the crawler and the scraper.

The crawler searches across the internet for the specific wanted data and collects the URLs of them in a queue whereas the scraper enters the URLs located by the crawler and scrape or extract the wanted data. The two components are important in extracting the needed data across the internet's web pages.

Security Concerns

- Data breaches: occur when unprotected data is inadvertently exposed during web scraping, leading to compromised information and legal risks.
- Malici Data and Denial of service attack: malic Data injection can harm data integrity, while denial of service attacks disrupt website performance.
- Intellec property Infringment: arises from unauthorized use of copyrighted content, posing legal risks.

Assess Common Security Risks, Threats, Attacks, and Vulnerabilities in an organization

1. Data Privacy Violations

2. Social Enginerring Attack

3. Price Scraping Attack

4. Content Scraping Attack

Existing methods of implementing the solution

- Accessing the needed data from the official website’ API if provided.
- Apply CAPTCHAS, as they allow legitimate users to complete a given task that is considered easily for humans as opposed to computerized machines.
- Monitor and detect the spiral growth of the website's volume or overload which indicates the movements of non-human activities being done to the website.
- A common method used by site owners is “Robots.txt” files, which are guidelines specified by the website itself in which it permits distinct parts of the website and disallows other parts.
- Implement a rate limiting algorithm that limits the frequency number of user requests.

Substitute Methods of Implementing the Solution

- Implement strong Firewall softwares onto the website that tracks and detects the movements of unauthorized bots or users or perhaps overload traffic that may come across the network.
- Upload sensitive and confidential data locally on the client's side instead of the server's side, this method will ultimately reduce the potential risk of extracting the data.
- Implement the usage of proxy which can assist in spinning the IP addresses by re-routing the scraper onto another IP address which helps in reducing the risks of data breaches and improve the efficiency.
- Modify the HTML periodically as the attackers usually access the most requested pages that are consistently available and written in clear sight.

Implementation

In this project we implemented a Python scraping tool called “facebook_scraper” to scrape comments from a post on Facebook.

Facebook’s policy to ensure non-malicious scraping:

- Adding a rate limit capping the number of times a user can interact with products in a given amount of time.
- Data limits which keep people from getting more data than they should need to use Facebook products normally.
- Blocking an account with abnormal activities specifically automated computer activities through pattern recognition

Implementation

Tools and Libraries:

1. **re (Regular Expressions):**

- A Python module which provides support for regular expressions.
- It provides functionalities such as pattern matching and manipulation of strings based on specified rules.

2. **csv (Handling .csv files):**

- A python module which is used for reading/writing CSV (Comma Separated Values) files.
- It provides functionalities to handle data in tabular format and is often used for database management.

3. **tkinter (Tkinter GUI Toolkit):**

- The Standard GUI (Graphical User Interface) toolkit for Python.
- It provides classes and functions to create graphical desktop applications with widgets like buttons, labels, entry fields, etc.

4. **facebook_scraper:**

- A Python library for scraping data from Facebook without needing an API key.
- It provides functionalities to scrape data from Facebook in various forms such as posts, comments, reactions, etc.
- Often used to simplify the process of accessing Facebook data programmatically for research, analysis and many other purposes.

Implementation

Code Overview

User Input

1. User enters post ID, maximum number of comments, CSV filename.

Validation

1. To ensure the post ID was provided in the right format.

CSV Filename Formatting

1. Appending a “.csv” extension in case the user doesn’t specify the time of file.

Scraping Process

1. “get_posts()” function from “facebook_scraper” library will be called with passing the following parameters: {post_urls = [post_id], options = {“Comments” : max_comments, “progress” : True}.

Implementation

Code Overview (Cont.)

CSV File Handling

1. A CSV file will be opened in write mode with UTF-8 encoding and the field names for the CSV file (‘Comment’, ‘Reply’) will be defined and written to the header row.
2. Each comment extracted will be iterated over and written under the ‘Comment’ column in the CSV file. Similarly for the replies and will be written under the ‘Reply’ column.

User Feedback

Case 1: If the scraping is successful, a message dialog will be displayed to inform the user that the comments have been successfully scraped and saved to the CSV file.

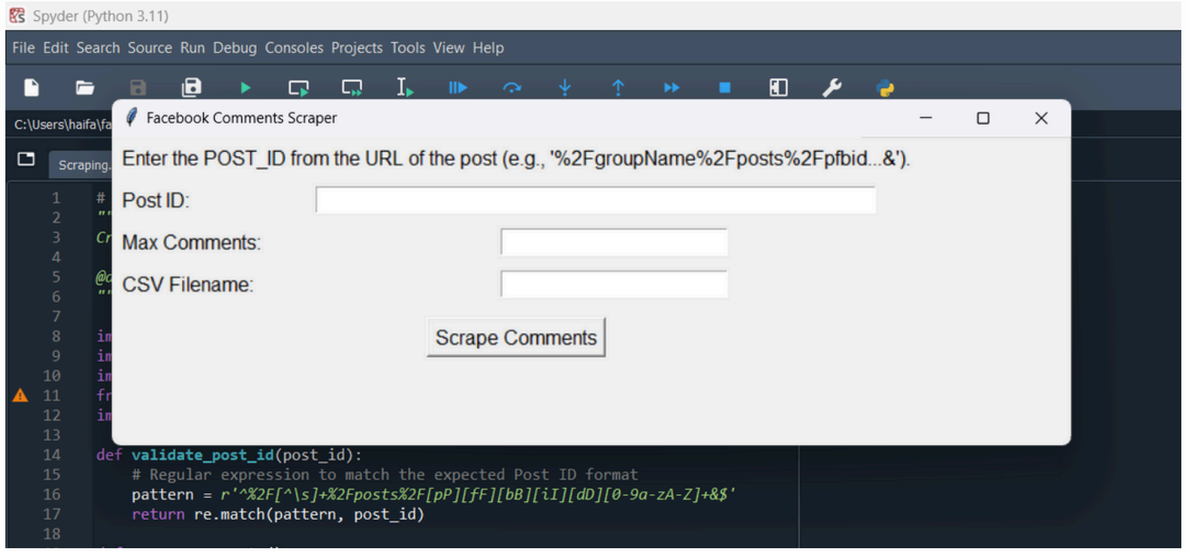
Case 2: If the scraping is unsuccessful, a message dialog will be displayed to provide the details of the error.

Implementation

Code Overview (Cont.)

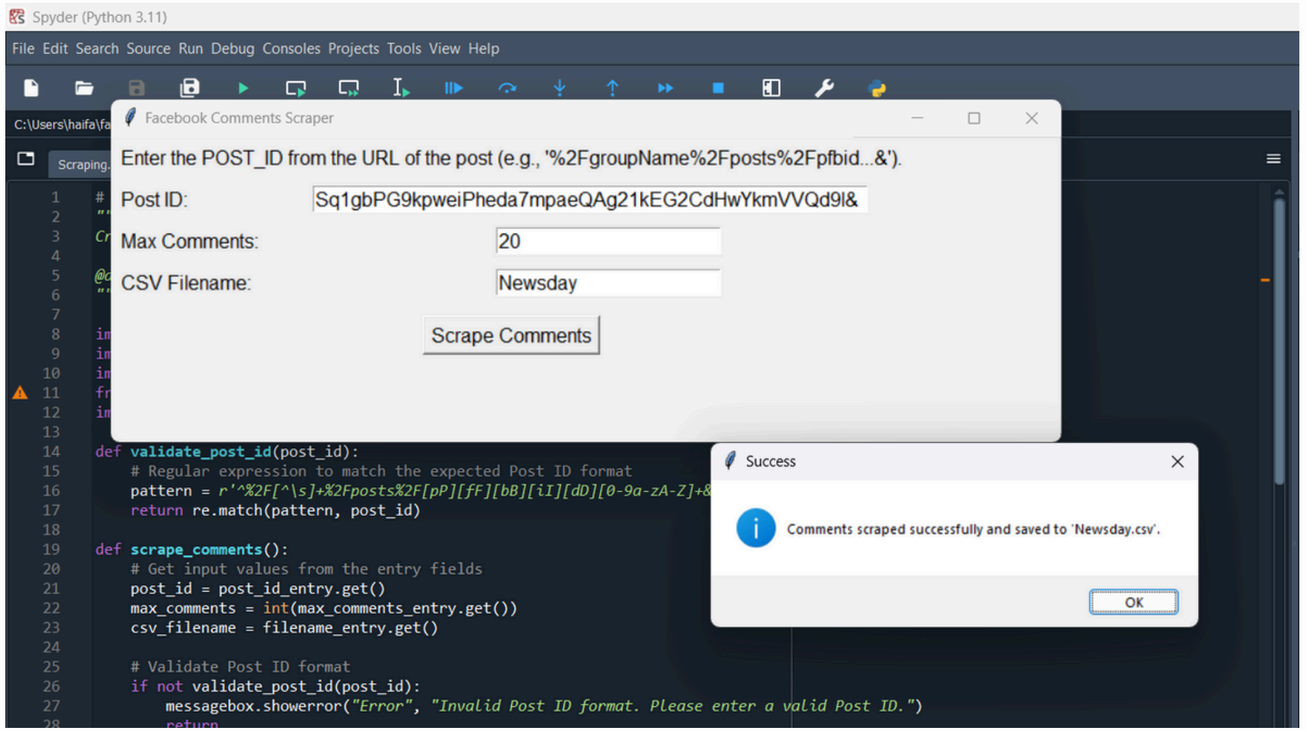
Tkinter GUI Interaction

1. Allows users to interact and enter the required input to initiate the scraping process.
2. Widgets such as entry fields for Post ID, maximum number of comments, and CSV file name, in addition to a button to start the scraping will be displayed inside the window.



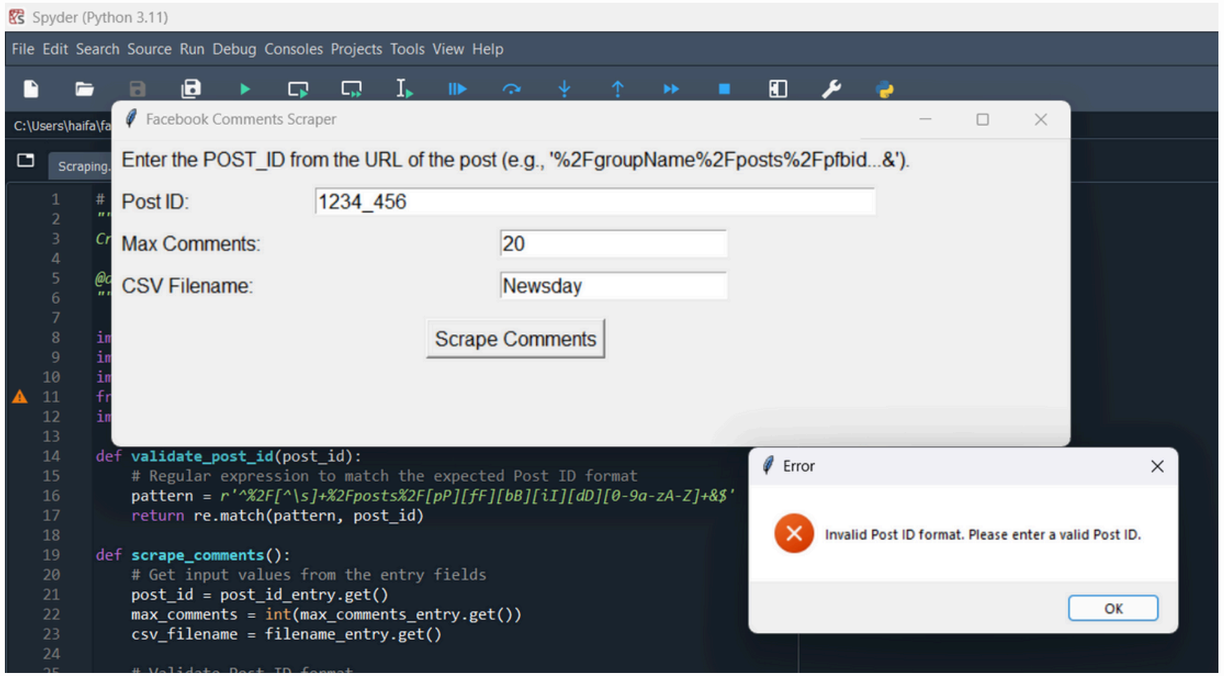
Implementation

Output Run Successful



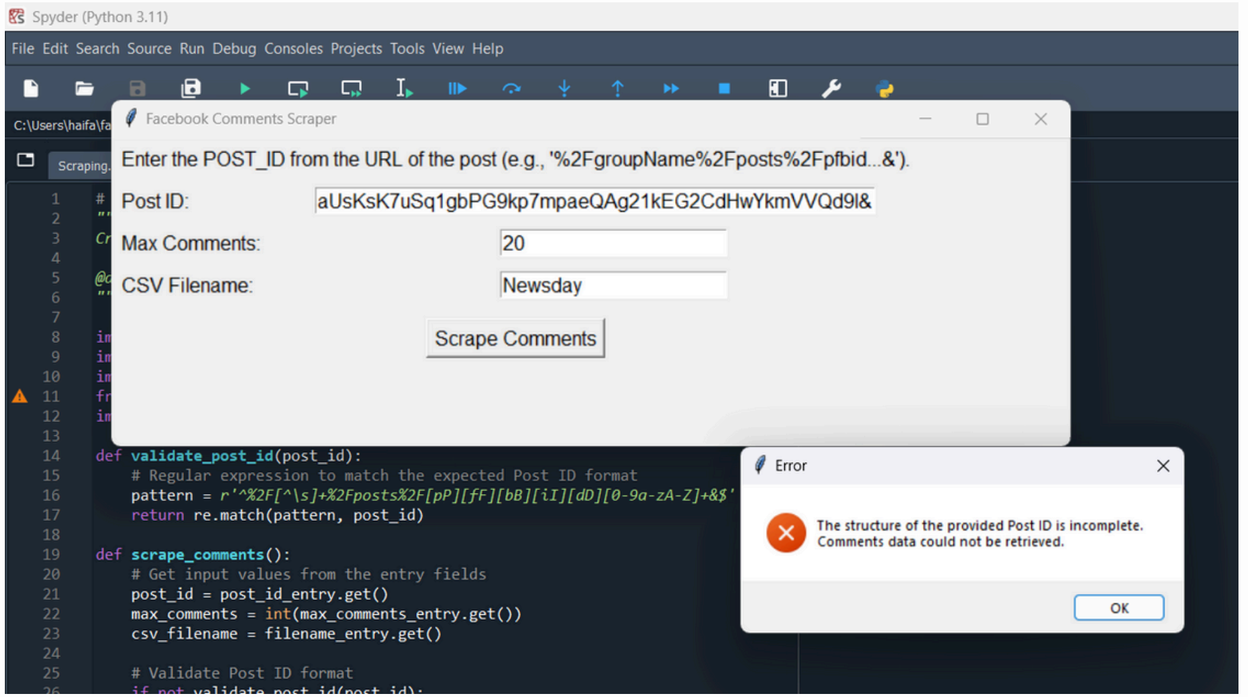
Implementation

Output Run Unsuccessful invalid post ID



Implementation

Output Run (Cont.) Unsuccessful missing post ID structure



Implementation

Complexity Analysis

1. **Input Parsing:** Takes constant time complexity **O(1)**.
2. **Validation:** Takes linear time complexity **O(n)**, where n is the size of the input Post ID string.
3. **Facebook Scraping:** Its complexity depends on the number of comments to scrape which is **O(m)**.
4. **Writing to a CSV:** Its complexity is proportional to the total number of comments and replies which is **O(c)**.
5. **Exception Handling:** Doesn't have a significance on the time complexity.

The Total runtime = O(1) + O(n) + O(m) + O(c), where **O(m)** is likely to be the most dominant term.

The complexity for scraping is heavily dependent on other factors such as network latency and response time from Facebook's servers.

Application and Recommendations to Overcome Threats

Maintaining the following recommended Facebook settings to prevent your data from unauthorized scraping possess great usage over the applications and recommendations to FaceBook web scraping.

- Navigate through your “Privacy Settings” in order to utilize your viewing options of who can see your public information or what is being shared.
- Walk through your “How People Find and Contact You” option on your settings page to utilize who can find your account through email and phone number.
- Utilize your basic information on your Facebook profile to ensure the visibility of you basic information.
- Customize your basic information on your Facebook page to control what is being shared with whom when posting.
- Follow up with the “Privacy Matters” page on Facebook for continuously informing yourself on the privacy initiatives at Meta Inc.

Conclusion

Throughout this project, we have explored web scraping to be a valuable software tool that helps in harvesting and collecting data from related websites and online resources which transforms it into structured datasets efficiently, which can be further used for various purposes as it can be a reason to support or oppose an infrastructure with the collected data; Moreover,in the literature review we have discusses several methods in overcoming the challenges that may overbare the scraping where a conducted security incident elaborated throughout the document which discusses how web scraping has been affecting several organizations negatively as well as positively for both sides. In addition, we have implemented a code on Python that demonstrates how web scraping is done in action on Facebook website while also implementing a graphical user interface (GUI) that demonstrates the scraping could be done in action as well.

Overall, it is important to protect and implement security tools and create a safe environment to provide confidentiality, integrity, and authentication.

Thanks!

Any Questions?