This script employs Selenium to automate the login to Twitter, retrieve recent tweets from designated usernames, tally mentions of a specified stock symbol, and repeat these steps at set intervals. Additionally, it incorporates error management and displays status updates and error notifications throughout its operation.

Overview

This script automates the process of logging into Twitter, fetching recent tweets from specified accounts, and counting the number of mentions of a given ticker symbol (e.g., $AAPL) in those tweets. The script runs in a loop, performing these actions at a specified interval.

Approach

1. **Login to Twitter:**
    - The script first logs into Twitter using provided credentials.
    - It navigates to the login page, enters the username and password, and submits the login form.
2. **Fetch Recent Tweets:**
    - After logging in, the script navigates to the specified Twitter accounts.
    - It waits for the tweets to load, collects them, and filters the tweets posted within the last 15 minutes.
3. **Count Mentions:**
    - The script then searches the collected tweets for mentions of the specified ticker symbol.
    - It counts the number of mentions and prints the results.
4. **Looping:**
    - The process repeats at a specified interval, allowing for continuous monitoring.

Difficulties Faced

Using BeautifulSoup with Twitter

- **Dynamic Content:** Twitter's content is dynamically loaded via JavaScript, making it difficult to scrape with static parsers like BeautifulSoup.
- **Authentication:** Logging into Twitter and accessing user-specific content is complex and often requires handling dynamic elements, which is not straightforward with BeautifulSoup.

Authentication with Selenium

- **Element Identification:** Locating elements for username and password input fields and the login button can be challenging due to frequent changes in Twitter's page structure.
- **Captcha and Security Measures:** Twitter may present captchas or other security measures, which require additional handling.

Overcoming Difficulties

- **Dynamic Content:** Using Selenium, which can interact with dynamically loaded content, allowed to effectively scrape tweets.
- **Element Identification:** Utilizing explicit waits and appropriate selectors ensured reliable interaction with login elements.
- **Manual Authentication:** Directly inputting credentials into the script facilitated the automated login process.

By using Selenium for web scraping and handling dynamic content, we successfully overcame the challenges posed by Twitter's structure and authentication mechanisms.