Crio.Do

Instagram Automation Tool

Project Description

Instagram is a great visual platform for people to share and view posts of other people around the world. A user will be doing basic activities like liking, sharing, commenting on a post, or following some other users. But often these rudimentary activities become quite repetitive and boring. So why not automate these processes? Automation using selenium is a great way to speed up this process and also save time.

Project Language(s)

Python

Difficulty

Beginner

Duration

15 hours

Prerequisite(s)

Python, HTML, CSS

Skills to be learned

Selenium, Automation, Python

Overview

Objective

You will be able to automate general activities like following, likes, comments, and exploring Instagram apps using python and selenium automation.

Project Context

Instagram is one of the leading social media apps today. You yourself must have had some experience in using Instagram. But often you might have got tired of following, liking, commenting on some person or some post every now and then. So why not automate the process using simple selenium automation techniques? Using Selenium web driver we can interact with a webpage like a real user and perform various actions like clicking, scrolling, typing to achieve goals like following, liking, and commenting (here).

Web automation today is a goto solution for testing an application, but it also has various other use cases like automating redundant processes for digital marketers and SEO specialists. Also, we can use automation to gather data for a particular business page, helping them with better user engagement by helping them figure out their audience's sentiment using NLP analysis on comments (challenge yourself by trying this out). For various computer vision models datasets are required. A good way to gather the data specific to the use case is by using automation rather than using the generic datasets on the web. This project can be a headstart for your data extraction journey. Use skills acquired in this project and build scripts for other websites as well.

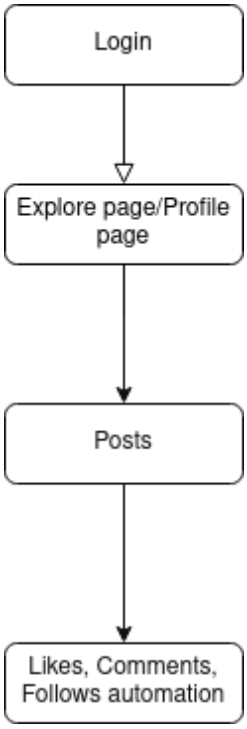
Modern websites dynamically load data which makes it hard to just make curl requests to that site, rather we need to interact with the page in order to extract the data. Apart from this, it is also really fun to build automation scripts for your daily web chores.

This project is a good start for beginners and a refresher for professionals who have dabbled in python scripts/selenium/web crawlers before. The experience of implementing this basic automation will be helpful in learning web crawlers and more, so feel free to innovate and explore!

Product Architecture

This Product Architecture consists of 5 stages as follows:

1. Login to your Instagram profile to begin
2. Redirecting to several URL endpoints like Explore page or tag pages
3. Redirecting to target interest pages or our own profile page
4. Traversing the images and storing metadata along the way
5. Using the metadata to fetch images, and use text data scraped for other sentiment analysis projects



Primary goals

- Automate various actions and interaction
- Storing various metadata regarding the profiles, pages, and posts we visit

High-Level Approach

- The main aim is to perform is to automate user interaction with Instagram
- Firstly we need to automate the login process, which includes entering username, password and clicking login button
- Next, we need a starting point to begin scraping, there are many choices for this, for example, explore page
- Now when we are at the explore page, we will try to go through the posts one by one and then perform a set of tasks, which are liking, commenting, following, and saving. To achieve these we will use selenium webdriver tools to perform browser interactions such as clicking, scrolling, and typing, etc.
- While going through posts we will save the image/video URL of these posts, and then we will consolidate these URLs to fetch the media and store it on our system.
- We will also store metadata like profile name, follower count, likes, comments, date posted, etc. to process them later for extended usage.
- Now that we have collected this metadata, we will use this data to analyze this text using NLP algorithms. We may also use the image fetched to train computer vision models. Note that this step is completely optional and beyond scope of this tutorial, but the idea for this is to motivate you and to build thought processes for data extraction.

The final automated process will be like this -

https://www.youtube.com/embed/JCzIKKEjNnw

Task 1

Setting up the environment and automate login into Instagram

First, we set up the environment and install the dependencies and do a low-level implementation (Proof of Concept) of the components involved in the project. For this, we are going to automate the login process to get started.

You may go through this tutorial as well for dependency installation. In this example, Firefox has been used as the browser but you are free to use any browser of choice; the only aspect that varies is the change in web driver packages for each browser.

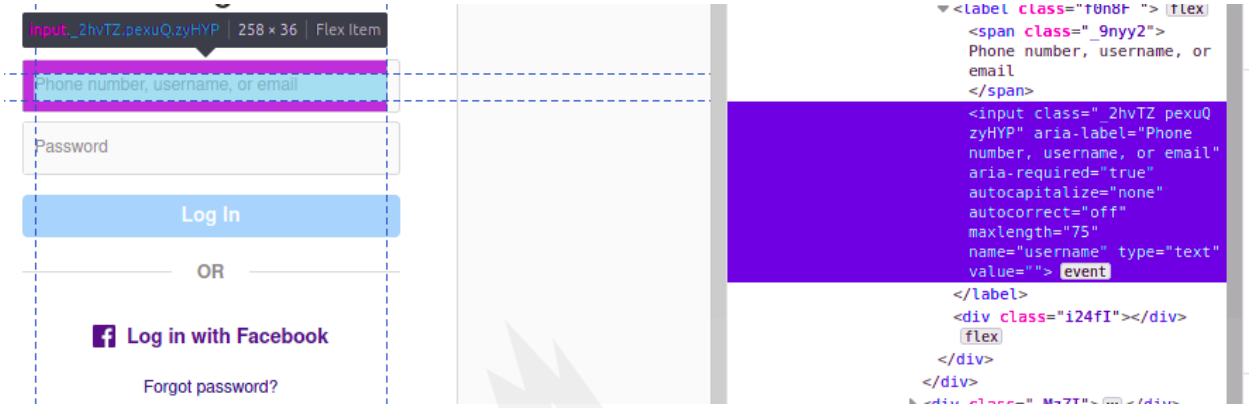
There are some nice to have settings like profiles and options you can go through in this documentation

Example:

```
options=Options()

options.add_argument("--allow-notifications")

options.add_argument("--allow-geolocation")

options.add_argument("--start-maximized")
```

Requirements

- Install Geckodriver and all other necessary packages
- Explore how Selenium webdriver and geckodriver works.
- Import all necessary libraries in the script.
- Open Instagram login page using driver. Use driver.get(url) to open Instagram login page.
- Checkout unique identifiers for the input fields login and passwords.
- Explore how xPaths help locating various elements in a web page.
- Examples:
    - xpath for following 'Username input field' element would be //input[@name='username']



- and xpath for the 'Save Post button' element will be //span[@class="wmtNn"]/div/div/button

- Discover login, password and submit elements and interact with them to achieve sign in. Use driver.find_element_by_xpath(xpath) to find the elements and use input_element.send_keys("text") to send text in the input boxes. Use submit_element.click() to achieve click action.

References

- Chrome dev tools
- Selenium
- Selenium with python
- Login using selenium

Note

- Knowledge of chrome dev tools is a must as you will be seeking and using various elements of a webpage.

Expected Outcome

The idea of this milestone is that you should explore how a web page is structured and how to inspect various components of a webpage like input fields, text, buttons etc. before interacting with them. Then to achieve automated login activity with just a few lines of code.

Task 2

Automating the process of crawling through posts on explore page

When we open the explore page we need to click on the first post to get started with the crawling. After that, we need to click on the like button to like the post and click on the next icon to advance to further posts. At this stage we need to store the URL of the images/videos in a metadata CSV along with other attributes like profile name, number of likes, and comments, etc, to be able to process it later. Further, if we need to comment as well, we can use the send_keys function in selenium to simulate typing by a user. Here we can store some hard-coded messages in an array in a json file to be loaded at the time of running and randomly publishing comments from this array.

Alternatively, we can also use ML to detect the sentiment of the post using the post description text/other comments to further improve it. But for basic proof of concept, we will stick to either hard-coded responses or duplicate other comments. Further, we can follow a profile using the following button. After following a page we can also redirect to that page and like other sets of posts on that page as well.

Now, to prevent thrashing and throttling at Instagram's servers, we need to limit our requests for that we need to put sleep of random time at various stages of our script. Also, instead of crawling just the explore page generically, we can also crawl particular tags as well.

Requirements

- Open various starter links for explore page/ tags page/ profile page using driver.get, note here while using driver.get function our page might reload and we can lose reference to previously-stored elements in variables.
- Explore element scroll and click functionality to interact with buttons. This might be needed for two reasons, one selenium can't interact with an element if it's not in the view, and secondly, in modern web pages content loads dynamically, so scrolling down may trigger loading of additional content. We can also achieve scrolling by running javascript functions using python as shown below.

```
driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")
```

- You may encounter some elements on which clicking directly is not supported. For those cases, we can use the lambda function as shown below.

```
follow_button = lambda: driver.find_element_by_xpath(xpath).click()
follow_button()
```

- Explore send keys function to achieve comments functionality. In selenium, we can use send keys functionality to emulate typing in a browser, which is quite useful to fill data in forms.
- During automation, we can also store some data like post URLs, text, and profile details for further processing in CSVs or a NoSQL database like MongoDB. Now for this step, we can structure the data gathering in a way that will help us in the next step while processing. It is preferred to use MongoDB in this case because the relations between profiles and posts and between posts and comments is a one-to-many relationship. But we can also store this data in csv as well in such a way that it is easier for further processing. We can create 3 csvs, one with all the profile data like handle, follower count, following count, no of posts, and profile URL/image. Another csv for posts which will contain for each post, no of likes, profile handle, media(image/video) url, the description of the post, date posted. We will also need one csv for storing comments as well. Now, the structure of storing data and storage type will depend on use cases and we can experiment with various permutations and combinations. This step will help you understand how databases are designed because we will be reverse engineering them in ways. So while you build scripts/scrapers for other websites as well, it will help you in understanding their database design as well.
- Check out some ML libraries or APIs to enhance engagement performance. This is a nice to have feature and requires skills beyond scope of this project. But if you are interested here are some ideas to try. We can use image processing to detect content in an image and customize our comments and replies. We can also use text analysis using NLP to also detect the intent of the comments/DMs and generate personalized comments/replies for them.

References

- Throttling
- NoSQL vs SQL
- NoSQL vs SQL pt.2

Expected Outcome

The idea of this milestone is that you should explore how to interact with the webpage and automate various interactions like clicking and typing which can be done using selenium.

Also, it will also help to grasp the concept of rate limiting and throttling and how to prevent it.

The automation process should be like this -

https://www.youtube.com/embed/JCzIKKEjNnw

Task 3

Saving the pictures/videos using links scraped in previous milestone and exception handling [OPTIONAL]

In this task, we will use urlretrieve functionality from urllib.request library to fetch posts from Instagram's server. While the automation is running it can encounter various exceptions for which we need to handle them using pythons, and this is called exception handling. We also need to log these errors to further develop the script.

The metadata (consisting of pictures, videos, comments mainly) are collected so that they can be subjected to some ML /DL models from which you can further advance this project by gaining analysis reports, like sentiment/emotion analysis, auto-reply, and much more.

NOTE This is a completely optional task and as it is just an idea for you to work on. If you are in for a challenge then surely you should try this out. Else you can simply skip this milestone.

Requirements

- Now we need to read URLs stored in CSV or the MongoDB database collected in previous milestones and explore various ways to request files from a server with those URLs. We can utilize the urlretrieve function from urllib.request library to make requests on Instagram servers.

```
urllib.request.urlretrieve(source_url, target_name)
```

- We can always store the data on our local machine, but often our storage may run out of space because image data is generally larger in size than text data. Instead of storing this data locally, we can choose to store them on the cloud, either in an AWS S3 bucket or gcp bucket. It is free at the starting because these services provide a good amount of storage in their free tier.
- It can so happen that urlretrieve may not fetch any result, which can happen due to several reasons like its expiration, or throttling, or due to unavailability, etc. For these cases, we may need to structure our code to handle it. In case of expiration/unavailability on the severs, we can store/mark those URLs with a flag. If the count of URLs with this flag as true continuously exceeds, we may need to trigger

to stop the script because this is the case or throttling. To prevent this from happening, we need to add sleep statements between each request.

- Now we have different kinds of data like Profile Stats, Image data, Text data. We can now build/model various machine learning/deep learning/NLP projects post this project. This step is optional as far as this project goes.
- Explore how to handle exceptions in python. The dynamic nature of modern websites can be prone to some unexpected events which might trigger some exceptions which may lead to the halting of the script. So, to handle these exceptions we might need to use exception handling to handle such cases.

```
try :
    //code which might throw an exception.
except :
    //what to do    if    an exception occurs. Hint - log it.
finally    :
    //this code runs always.
```

- Explore how to log these errors for continuous development. Logging is also used to monitor the health of the script which is generally running without supervision. So we need to add Log statements for various events in our script. That is why you might want to put log statements at various stages of automation.

References

- Download files using python
- Logging in python
- Sentiment analysis
- Image classification model for Dog vs Cat
- Image classification model for Dog vs Cat pt.2

Expected Outcome

The idea of this Milestone is that you should explore how to make requests to the server to fetch the data and how to handle and log exceptions in python

Task 4

Publish to GitHub

Publish your project by making a new GitHub repository and have some green goodness!

Requirements

- Kindly go through this Byte if you're unfamiliar with the Git workflow.

References

- How to add an existing project to Github

Expected Outcome

You should be able to deploy the application on a cloud platform.

Task 5

Deploy the application on a cloud platform

Now that your application is complete, it's ready to be deployed! Go on and deploy your application on the Google Cloud Platform in a Docker Container.

If you are new to cloud services, you can go through the QPrep - System Design micro-experience available on the platform before proceeding. Also, if you're new to Docker, kindly go through the Docker Introduction and Docker Advanced Bytes.

Requirements

- Create a Docker container for your application. It will make the deployment easier.
- Setup a cloud instance on GCP and activate it.
- Upload your files to the platform. You should simply use your GitHub repository here since it'll do the job by a simple git clone .
- Run your application on the platform.
- Here you might want to run the automation in headless mode.

```
options.headless=True
```

Note

- You are free to use any Selenium supported cloud provider.

References

- Docker — Beginner's Guide
- Google Cloud
- Google Cloud Platform For Beginners - Mayank Chourasia - Medium

Expected Outcome

You should be able to deploy the application on a cloud platform.