# CHAPTER 4

# Crawler Design and Implementation

In this chapter, a web crawler is created using the Selenium Python package to programmatically scrape some web pages from the Reddit web forum. Selenium is a Python package or a tool that can help to crawl the website to obtain your interested data (Muthukadan 2011). So, there will be four (4) sections involved in this chapter to explain the experiment completed. The four sections are as follow:

- Section 4.1: Introduction
- Section 4.2: Web Crawler Framework
- Section 4.3: Web Crawler Functions
- Section 4.4: Data Sources of Web Crawler.

## 4.1 Introduction

A web crawler is also known as a web spider where it is a computer program that can automatically explore the World Wide Web and the main objective of creating it is to acquire the web data, web content and web structure of a website. It can navigate the information on the webpage on its own, and the search engine is inextricably linked to the web crawler. Besides, its most significant duty is to crawl through the massive amounts of data on the Internet, find useful information, and then store it in a local database. A web crawler begins with a list of URLs to visit, called the seed. The crawler will then search all the links in the HTML for each URL, filters those links based on certain criteria, and adds the new links to a queue. All of the HTML or some specific information is extracted and to be processed.

## 4.2 Web Crawler Framework

The web crawler framework is divided into four (4) layers, which is Chrome Driver Activation, Link Extraction, Post Content Extraction and Store Post Content. *Figure 4.2* is an illustration of the web crawler framework.

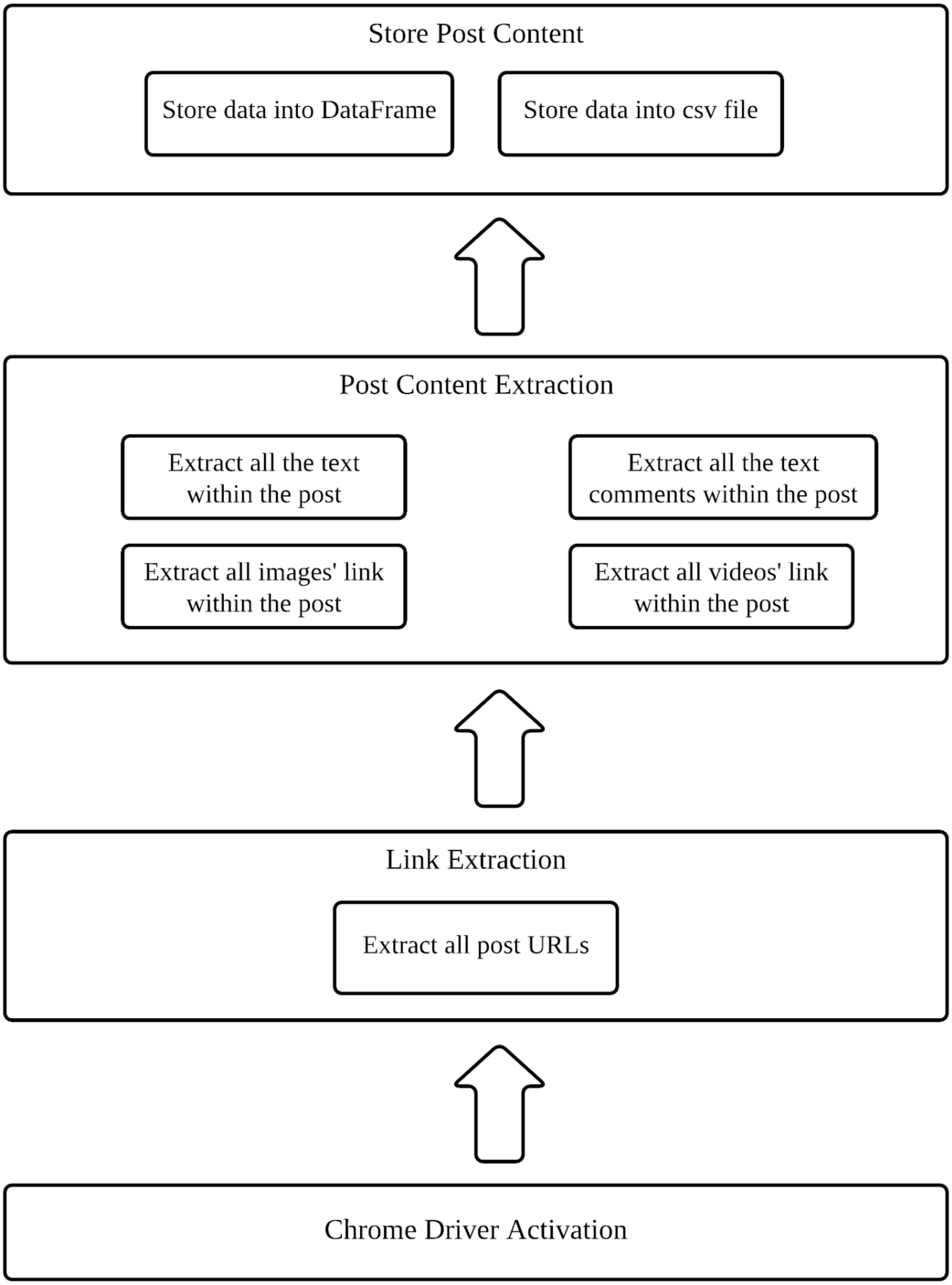
Store Post Content
Store data into DataFrame
Store data into csv file
Post Content Extraction
Extract all the text within the post
Extract all the text comments within the post
Extract all images' link within the post
Extract all videos' link within the post
Link Extraction
Extract all post URLs
Chrome Driver Activation

*Figure 4.2.1: The Star Web Crawler Layered Framework*

### 4.2.1 Chrome Driver Activation

First of all, the Chrome driver is chosen to perform browser automation. Launch the Chrome browser by passing in the sub-forum URLs of the Reddit web forum page that you want to crawl. For example, we can pass in this 'badminton' sub-forum URL to crawl the data contain in this sub-forum, https://www.reddit.com/r/badminton/.

### 4.2.2 Link Extraction

The crawler now will extract each post's URL within the sub-forum. In my case, the crawler now will extract the URL of each post within the sub-forum called 'badminton'.

### 4.2.3 Post Content Extraction

In this phase, all the extracted URLs are used to retrieve the post content. The web crawler now will extract all the text within the post. Aside from crawling the text within the post, the crawler also can crawl the images and videos as well if the post contains images or videos. The crawler will extract the particular image link and video link. On top of that, the crawler will also crawl all comments made by other users in the comment section of the post. In order to retrieve or extract data from the webpage, I would use web element locating techniques like CSS selector and xpath to locate to all the HTML tags that I want to extract and then extract the data within it. Some examples of the HTML tags I have located to are <div>, <a>, <p>, <img> and so on.

### 4.2.4 Store Post Content

Finally, the data crawled will be stored into DataFrame and then saved into a CSV file named *badminton.csv*.

## 4.3 Web Crawler Functions

Two (2) functions or methods were created in this web crawler to assist in crawling Reddit content: the *user_login* function and the *collect_subData* function.

1. `user_login()`

| | |
|---|---|
| **Description** | This function will make the crawler to login into the user account by accessing the *username* and *password* entered by the user in the .env file. |
| **Example** | `user_login()` |

2. `collect_subData(number, postlinkaddress, post_lst_details)`

| | |
|---|---|
| **Description** | This function will accept a post URL and extract all the content in the post. After that, all of the extracted results will be saved as a dictionary and appended to a list. |
| **Argument (**`number, postlinkaddress, post_lst_details`**)** | `number` is a counter that helps to aggregate the total number of processed posts. |
| | `postlinkaddress` is a post URL link that would be used by this function to extract its content. |
| | `post_lst_details` is a list that is used to store or appended extracted results of each post. |
| **Example** | `collect_subData(number, postlinkaddress, post_lst_details)` |

## 4.4 Data Sources of Web Crawler

In this project, the crawler will focus on the web forum pages of Reddit where this Reddit website has a lot of different sub-forums (known as *subreddit* in Reddit) like *badminton*, *UncleRoger*, *anime*, *politics, MalaysianFood* and many more. Therefore, all sub-forums on Reddit serve as data sources for the web crawler. However, due to there are too many sub-forums, I will only focus on crawling *badminton* and *UncleRoger* sub-forums (also known as *subreddit* in Reddit) in this project.

## 4.5 Chapter Summary and Evaluation

In summary, this chapter explains a web crawler's logic flow and its ability to automatically browse World Wide Web content for image or text retrieval. All the retrieved text can be used for training to carry out different projects in the NLP domain such as sentiment analysis, text generation, AI Chatbot and so on. Meanwhile, all the retrieved images also can be used for training to carry out different projects in the Image Processing domain such as face recognition, object detection, image classification and many more. Most importantly, many businesses have discovered the value of data in assisting them to grow their businesses in today's globe. As a result, many companies have begun to hire data scientists that can help them to crawl and process all the collected data.