These are the instructions which I followed to perform Web scraping on the given URLS.

1. Mount Google Drive:

I used Google Colab so I have connected Google Drive connected to Colab. The code drive.mount('/gdrive') will mount your Google Drive.

1. Change the current directory:

The line %cd '/gdrive/MyDrive/project/ TestAssignment' changes the current directory to the specified path in your Google Drive. Adjust the path according to your project's directory structure.

1. import necessary packages:

The required packages are imported using import statements. Make sure you have these packages installed before running the code.

1. Read input data:

Ensure you have an Excel file named "Input.xlsx" in the specified directory. The code df = pd.read\_excel('Input.xlsx') reads the Excel file and stores the data in a pandas DataFrame named df. The file should contain columns named "URL" and "URL\_ID".

1. Data Extraction from URLs:

The code iterates over each row in the DataFrame using a for loop and performs the following steps for each URL:

Sends a GET request to the URL using requests.get(). Make sure the URLs in the "URL" column are valid and accessible.

Creates a BeautifulSoup object from the response content using the HTML parser.

Extracts the title from the HTML using soup.find('h1').get\_text().

Extracts the text content from paragraph tags (<p>) in the HTML.

Writes the title and article text to separate text files in the specified directory.

Set Directories: The code defines three directory paths: text\_dir, stopwords\_dir and sentment\_dir. Adjust these paths based on your project's directory structure.

1. Load Stopwords:

The code reads stop words from files in the stopwords\_dir directory and adds them to a set named stop\_words.

1. Tokenize and Filter Text:

The code reads text files from the text\_dir directory, tokenizes the text using nltk.tokenize.word\_tokenize(), removes stop words from the tokens and adds the filtered tokens to a list named docs.

1. Load Positive and Negative Words:

The code reads positive and negative words from files in the sentment\_dir directory and adds them to sets named pos and neg, respectively.

1. Calculate Sentiment Scores:

The code iterates over each document in docs and calculates positive score, negative score, polarity score, and subjectivity score based on the presence of positive and negative words.

1. Measure Text Metrics:

The code defines a function measure() that takes a text file as input, measures various text metrics such as average sentence length, percentage of complex words, and Fog Index. It iterates over each text file and calculates these metrics, storing them in separate lists.

1. Perform Text Metrics Calculation:

The code iterates over each text file and calculates metrics such as word count, average word length, and count of personal pronouns.

1. Read Output Data Structure:

Ensure you have an Excel file named "Output Data Structure.xlsx" in the specified directory. The code output\_df = pd.read\_excel('Output.xlsx’)