*#Dependencies*

from bs4 import BeautifulSoup as bs

import os

import requests

import pymongo

from webdriver\_manager.chrome import ChromeDriverManager

*# Create a Beautiful Soup object*

soup = bs(html, 'html.parser')

type(soup)

*html(can be obtained through a file(through filepath = os.path.join("..", "Resources", "template.html")*

*with open(filepath) as file:*

*html = file.read()*

*or through url using:*

*response = requests.get(url)*

*soup = BeautifulSoup(response.text, 'html.parser')*

*# Print formatted version of the soup*

print(soup.prettify())

*# Extract the title of the HTML document*

soup.title

*# Extract the text of the title*

soup.title.text

*# Clean up the text*

soup.title.text.strip()

*# Extract the contents of the HTML body*

soup.body

*# Extract the text of the body*

soup.body.text

*# Text of the first paragraph*

soup.body.p.text

*# Extract all paragraph elements*

paragraphs=soup.find\_all('p')

for paragraph in paragraphs:

print(paragraph.text)

*# Extract paragraph by index*

soup.body.find\_all('p')[0]

# The text of the first paragraph

soup.body.find('p').text

# results are returned as an iterable list

results = soup.find\_all('li', class\_="result-row")

# Loop through returned results

for result in results:

# Error handling

try:

# Identify and return title of listing

title = result.find('a', class\_="result-title").text

# Identify and return price of listing

price = result.a.span.text

# Identify and return link to listing

link = result.a['href']

# Print results only if title, price, and link are available

if (title and price and link):

print('-------------')

print(title)

print(price)

print(link)

except AttributeError as e:

print(e)

# Loop through returned results

for result in results:

# Retrieve the thread title

title = result.find('p', class\_='title')

# Access the thread's text content

title\_text = title.a.text

# print(title\_text)

try:

# Access the thread with CSS selectors

thread = result.find('li', class\_='first')

# The number of comments made in the thread

comments = thread.text.lstrip()

# Parse string, e.g. '47 comments' for possible numeric manipulation

comments\_num = int(comments.split()[0])

# Access the href attribute with bracket notation

link = thread.a['href']

# Run if the thread has 20 or more comments

if (comments\_num >=20 ):

print('\n-----------------\n')

print(title\_text)

print('Comments:', comments\_num)

print(link)

except AttributeError as e:

print(e)

# Initialize PyMongo to work with MongoDBs

conn = 'mongodb://localhost:27017'

client = pymongo.MongoClient(conn)

# Define database and collection

db = client.craigslist\_db

collection = db.items

# Examine the results, then determine element that contains sought info

# results are returned as an iterable list

results = soup.find\_all('li', class\_='result-row')

# Loop through returned results

for result in results:

# Error handling

try:

# Identify and return title of listing

title = result.find('a', class\_='result-title').text

# Identify and return price of listing

price = result.a.span.text

# Identify and return link to listing

link = result.a['href']

# Run only if title, price, and link are available

if (title and price and link):

# Print results

print('-------------')

print(title)

print(price)

print(link)

# Dictionary to be inserted as a MongoDB document

post = {

'title': title,

'price': price,

'url': link

}

collection.insert\_one(post)

except Exception as e:

print(e)

# Display items in MongoDB collection

listings = db.items.find()

for listing in listings:

print(listing)

# scrape the datetime

datetime = result.find('span', class\_='article-item\_\_date')['data-date']

# get only the date from the datetime

date = datetime.split('T')[0]

# Setup splinter

executable\_path = {'executable\_path': ChromeDriverManager().install()}

browser = Browser('chrome', \*\*executable\_path, headless=False)

url = 'http://quotes.toscrape.com/'

browser.visit(url)

for x in range(1, 6):

html = browser.html

soup = BeautifulSoup(html, 'html.parser')

quotes = soup.find\_all('span', class\_='text')

for quote in quotes:

print('page:', x, '-------------')

print(quote.text)

browser.links.find\_by\_partial\_text('Next').click()

# Optional delay for loading the page

browser.is\_element\_present\_by\_css('div.list\_text', wait\_time=1)

browser.quit()