CS 410 Final Project Write-up

Analyze Real Estate Trends During Pandemic by Utilizing Web Crawler

I. Project Overview

A web crawler is built to grab home listing/sales related data from www.realtor.com. This website has national home listings in the USA. By analyzing the real estate dataset that was extracted from the internet, I will provide a summary report of the real estate trends and listing info during the pandemic. The tools used in this project will be Python, MS SQL Server, and Power BI.

This project contains three sections, first is to build a web crawler, second is to clean the data, and third is to build data visualizations. This project is focusing on the first section, building a web crawler, which is more related to this course.

I. Team information

Team Name: Team-Chuan Jiang
Member: Chuan Jiang (NetID: chuanj2)

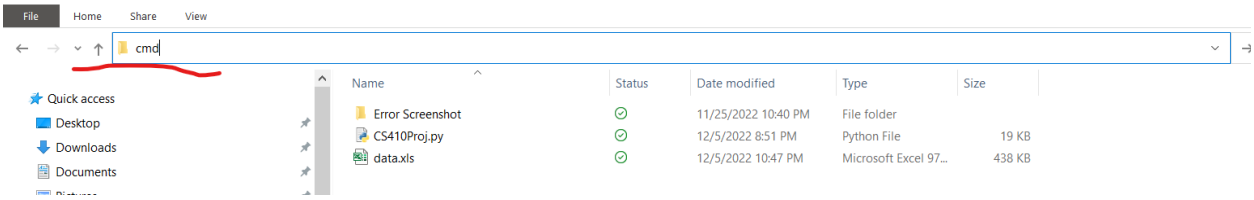
II. Requirements and steps to run the code

My testing environment: Window 10 with python version 3.10.0 installed. Codes should also work with other python 3 versions.

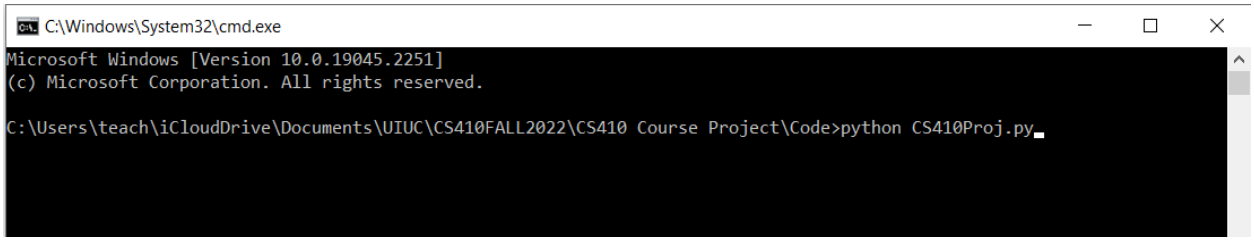
1. Download CS410proj.py to your local folder.

2. Installed required Python libraries if do not have these installed: xlwt, requests, easygui
   a.) You can use pip to install them, ex: pip install xlwt

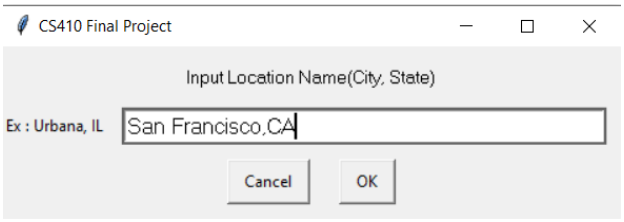
3. Go to the local folder where it saves CS410proj.py. Enter cmd in the folder address bar to open the terminal



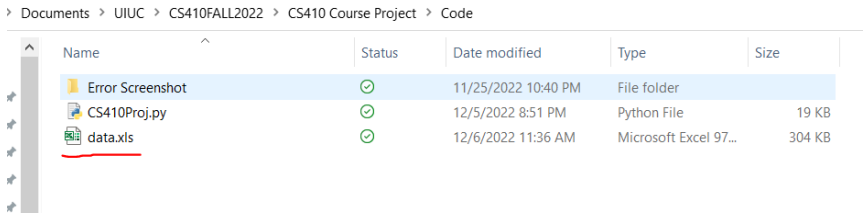
4. Enter command 'python CS410proj.py' to run the code



5. You will see a pop-up window like below, please enter City, State here, then click OK. Ex: San Francisco, CA



6. An excel worksheet 'data.xls' is created under the same folder.



Open the excel worksheet, it contains the data we just scraped from www.realtor.com.

	A	B	C	D	E	F	G	H	I	J	K
1	price	beds_num	baths_num	sqft	list_date	type	company_name	street	city	state	postal_code
2	1239000	2	2	1027	2022-12-06	condos	Coldwell Banker Realty	300 Berry St Unit 408	San Francisco	California	94158
3	1249000	3	2	1582	2022-12-06	single_family	Prime Metropolis Prop., Inc.	738 Geneva Ave	San Francisco	California	94112
4	363324	1	1	631	2022-12-06	condos	Kw Peninsula Estates	1400 Mission St Apt 710	San Francisco	California	94103
5	1449000	2	3	1655	2022-12-06	condos	Sotheby's International Realty - San Francisco Brokerage	415 Bryant St Apt 7	San Francisco	California	94107
6	1344888	9	3	2664	2022-12-05	townhomes	Luxe Places International Real	2924-2926 Cesar Chavez St	San Francisco	California	94110
7	699000	1	1	626	2022-12-05	condos	Sequoia Real Estate	1160 Mission St Unit 2010	San Francisco	California	94103
8	2190000	2	2	1396	2022-12-05	condos	Mosaik Real Estate	690 Market St Unit 2202	San Francisco	California	94104
9	779000	1	1	665	2022-12-05	condos	Vanguard Properties, Inc.	286 Parnassus Ave	San Francisco	California	94117
10	6000000	3	5	2780	2022-12-05	condos	COMPASS	1170 Sacramento St Apt 12D	San Francisco	California	94108
11	2739000	3	3	1833	2022-12-05	single_family	KW Advisors	4277 23rd St	San Francisco	California	94114
12	795000	1	2	1164	2022-12-05	condos	COMPASS	77 Dow Pl Apt 110	San Francisco	California	94107
13	1298000	2	2	1167	2022-12-05	condos	Bay Real Estate Group	219 Brannan St Unit 5D	San Francisco	California	94107
14	1499000	3	2	1200	2022-12-05	single_family	RedFin	51 Brewster St	San Francisco	California	94110
15	595000	2	1	670	2022-12-05	single_family	Coldwell Banker Realty	383 Haight St	San Francisco	California	94102
16	2995000	4	4	3066	2022-12-04	single_family	Concolors Real Estate	335 30th Ave	San Francisco	California	94121

Notes: For Mac users, you should follow the above steps by using mac terminal. (Not tested)

IV. Used Libraries/Models

Three libraries used in this project: xlwt, requests, easygui

xlwt: a library for writing data and formatting information to older Excel files

easygui: provides an easy-to-use interface for simple GUI interaction with a user. It does not require the programmer to know anything about tkinter, frames, widgets, callbacks or lambda. It runs on Python 2 and 3, and does not have any dependencies.

Requests: HTTP library for Python. Requests allows you to send HTTP/1.1 requests extremely easily. There's no need to manually add query strings to your URLs, or to form-encode your POST data. Keep-alive and HTTP connection pooling are 100% automatic.

V. Code Structure

The structure of this web crawler is simple. All codes are under python file CS410Proj.py.

In this file, there are three different parts: header and cookies, data extraction, and data load.

Header and cookies: this part handles the communication between user and server. Here is between the web crawler and server.

Data extraction: handles how to extract the data, and what data needs to be extracted.

Data Parsing and load: Handles in what kind of format of the extracted data should be written into the destination.

VI. Conclusions and final thought

There are some steep learning curves there if I would like to handle and dig into the complicated blocking and captchas. For example, I tried to scrape Zillow.com, which is another national real estate platform. They have much stronger anti-bot captchas. First, I tried to use the Requests library to scrape the data, I found that the parameters set up is more way more complicated and may result in blocking, then I used Selenium library (a Python library used for automating web browser to do a number of tasks such as web scraping), the Zillow.com will scan the bot and requests a human to solve the captcha, which is an automated algorithm-generated textual and visual. I spent time and researched how to handle the captcha, it included invariant recognition (identifying different shapes, images of the same alphabet, object), segmentation (identifying overlapping alphabets), and parsing context (holistically understanding the image, text, or audio) [3]. It was far beyond the scope of this project and required much more time to study and test.

I changed to scrape data from realtor.com. I started with Requests library to scrape the data, the challenge was to set up different parameters to scrape the corrected data which includes page, offset, city location and data parsing. Eventually, I was able to scrape the data I wanted without being blocked by realtor.com.

VII. Referenced Resources

[1] How to Grab HTTP Headers and Cookies for Web (Scrapinghttps://www.scraperapi.com/blog/headers-and-cookies-for-web-scraping/)

[2] Zillow Data Scraping using Python | Scrape Real Estate Listings (https://www.youtube.com/watch?v=pzptMqULnyE&t=354s)

[3] How To Solve CAPTCHA While Web Scraping? (https://medium.com/dataseries/how-to-solve-captcha-while-web-scraping-9335c95800eb)

[4] How to Scrape Zillow Real Estate Property Data in Python (https://scrapfly.io/blog/how-to-scrape-zillow/)