CIS 612 Big Data and Parallel Processing

LAB-3:

REVIEW EXTRACTION FROM YELP RESTAURANTS

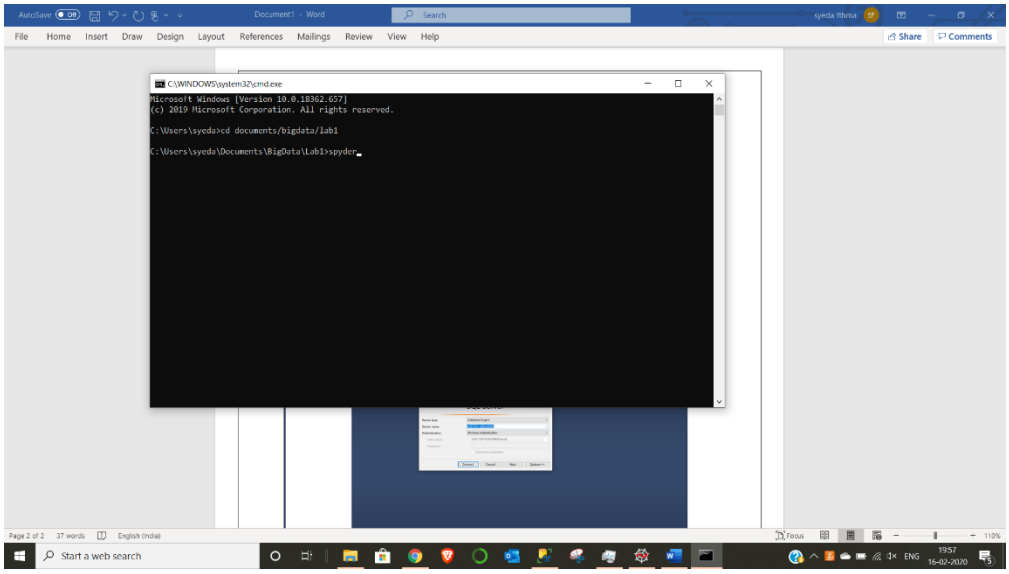
Fathima Syeda

CSU ID-2790024

Platform Setup:

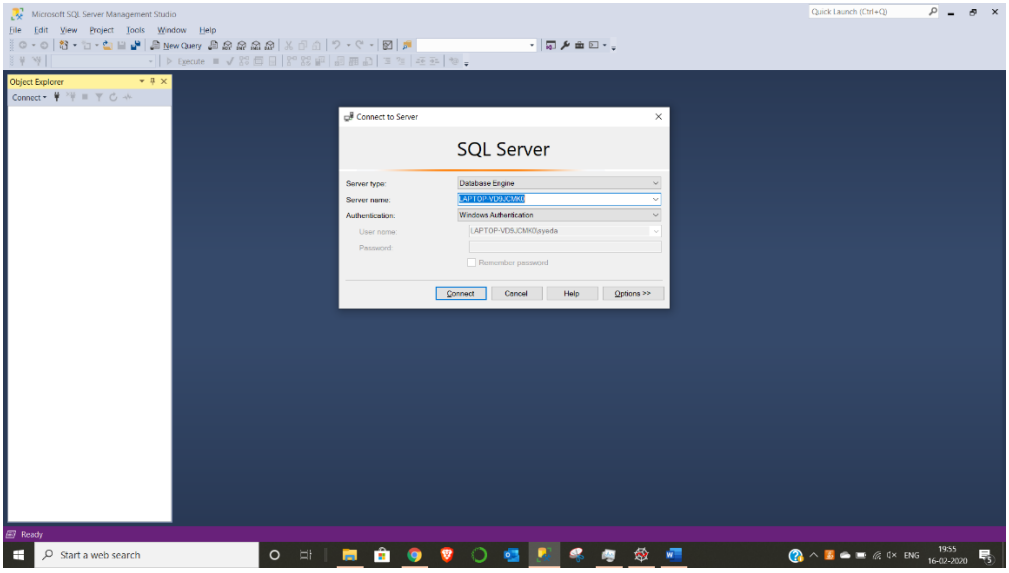
This lab was done using Microsoft SQL Server Management Studio and Python.

1) Spyder IDE was used to write the python code. In order to open Spyder you go to the command prompt and go to the directory you wish to save your source code in and then type Spyder to launch the Spyder IDE.



Once Spyder opens, you start typing your python source code.

2) The server can be started by opening the Microsoft SQL Server Management Studio application.



Once it opens you may make note of your server's name i.e. "HASNAIN2020 " (which will be used in the code)and press connect to connect to it.

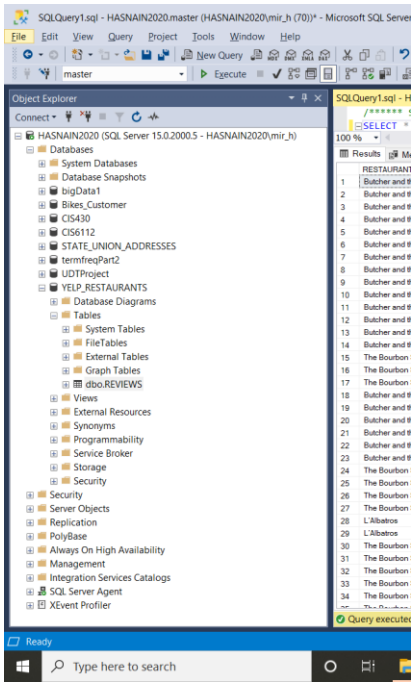
Program Outline:

The program makes use of python's lxml, pyodbc, urllib.parse, os libraries.

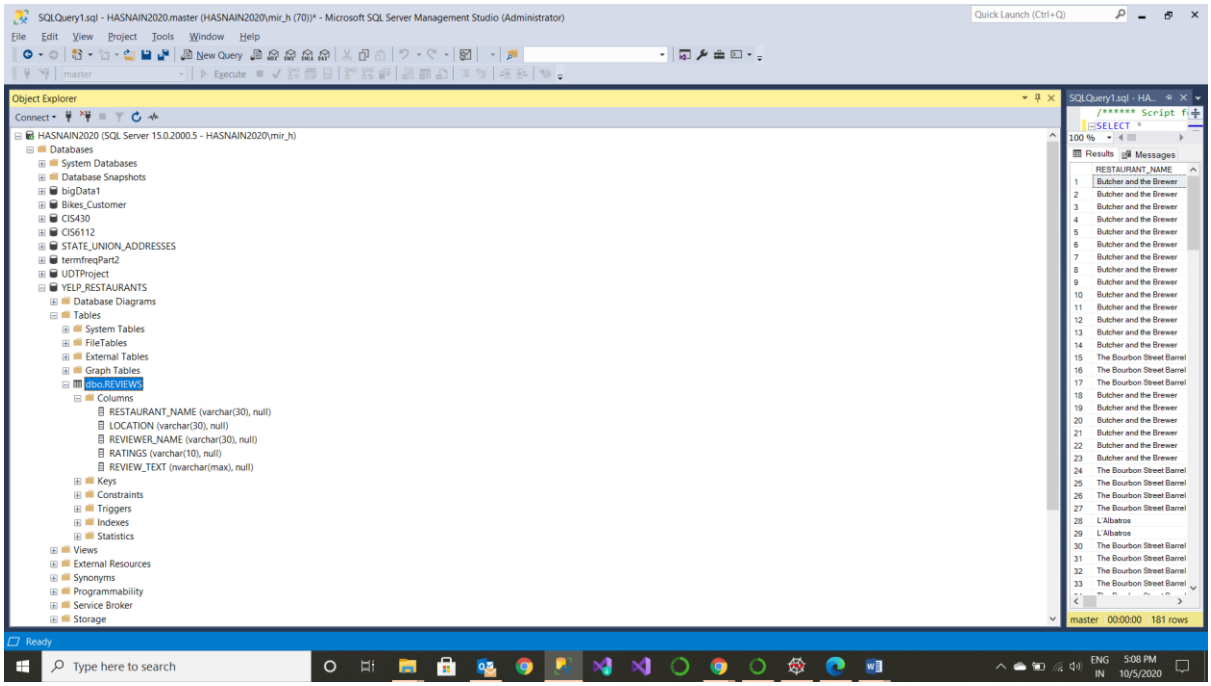
The program consists of five functions: main, connecttoserver(), insert_into_rows(), extract_info(link), write_to_file(resname,file_name,text), makeDirectory(resName).

We first make sure that the SQL server is still connected and running when we had connected to it in the setup part. Then we comeback to Spyder and run the python program.

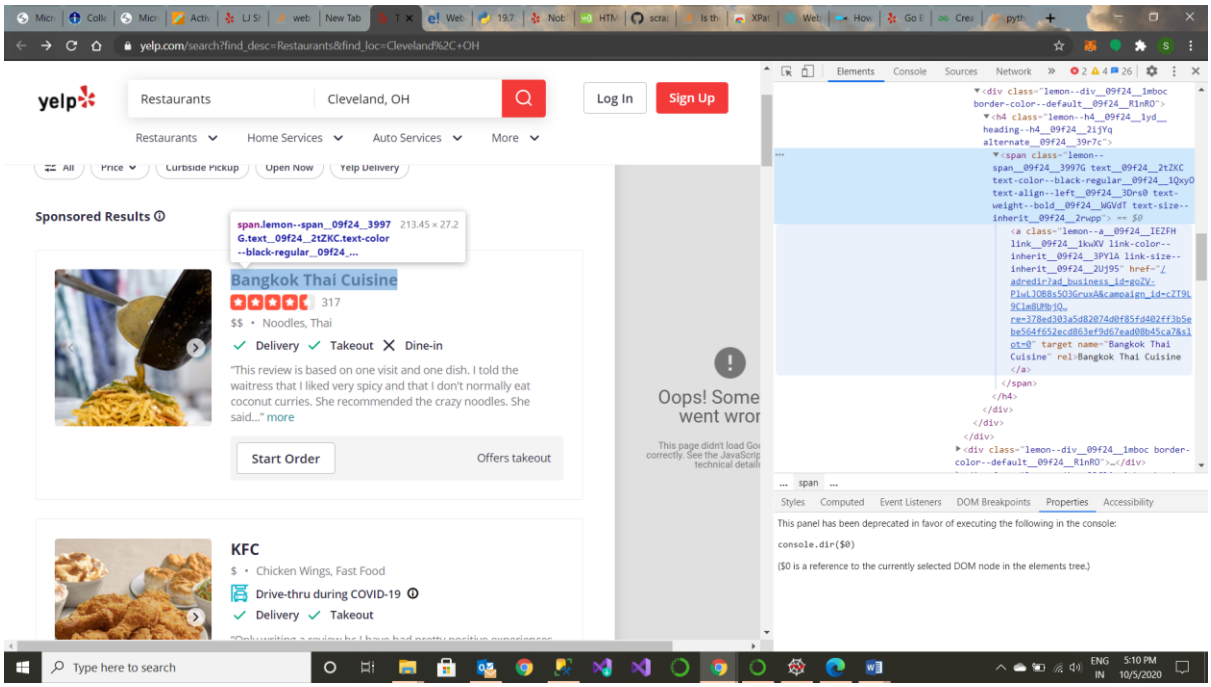
A new database called YELP RESTAURANTS with a table called dbo.REVIEWS in it.

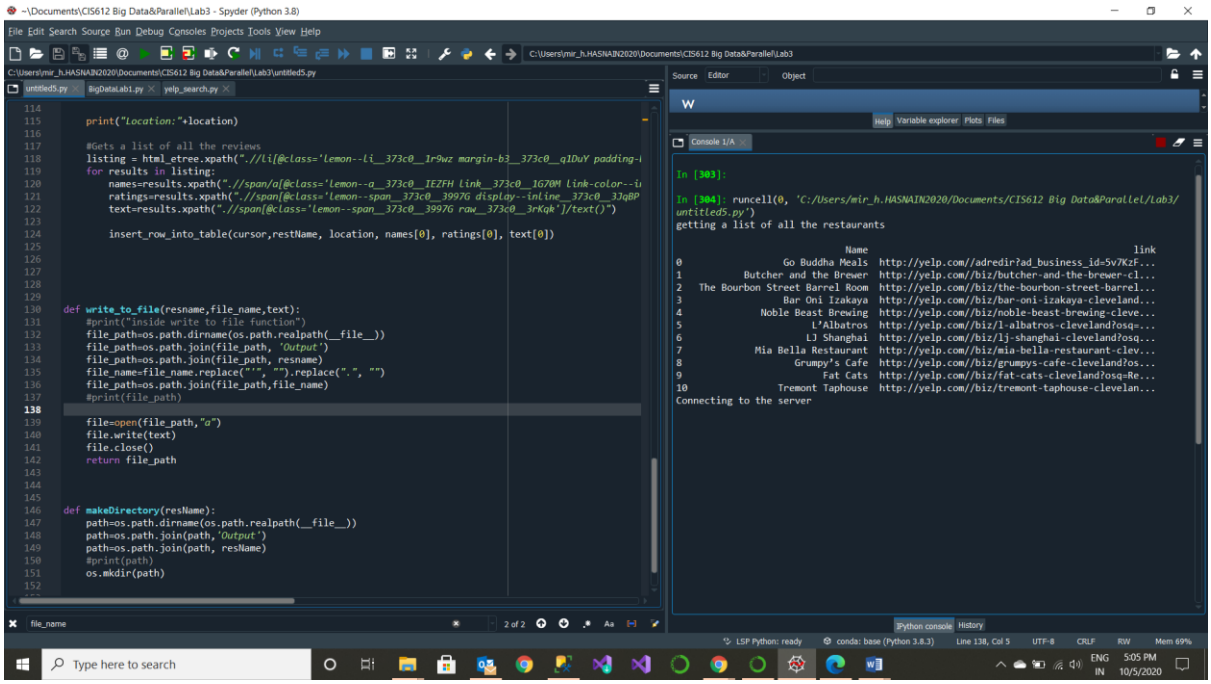


The table has columns- restaurant Name, location, Reviewer names, ratings, text

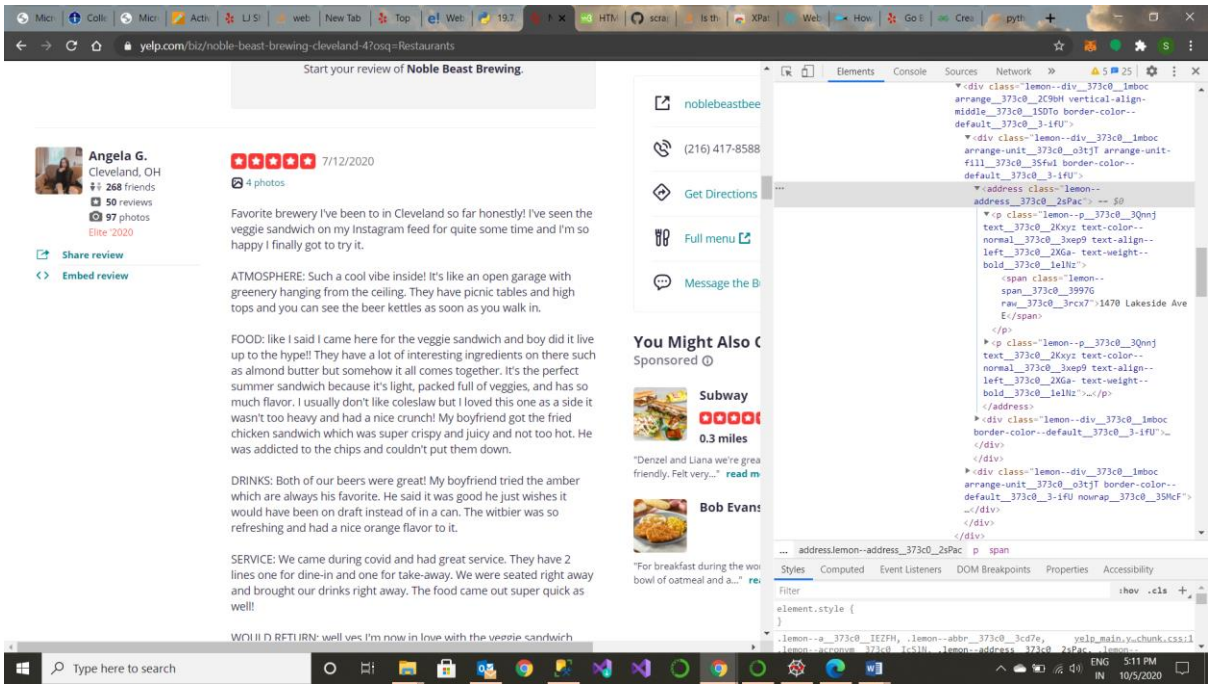


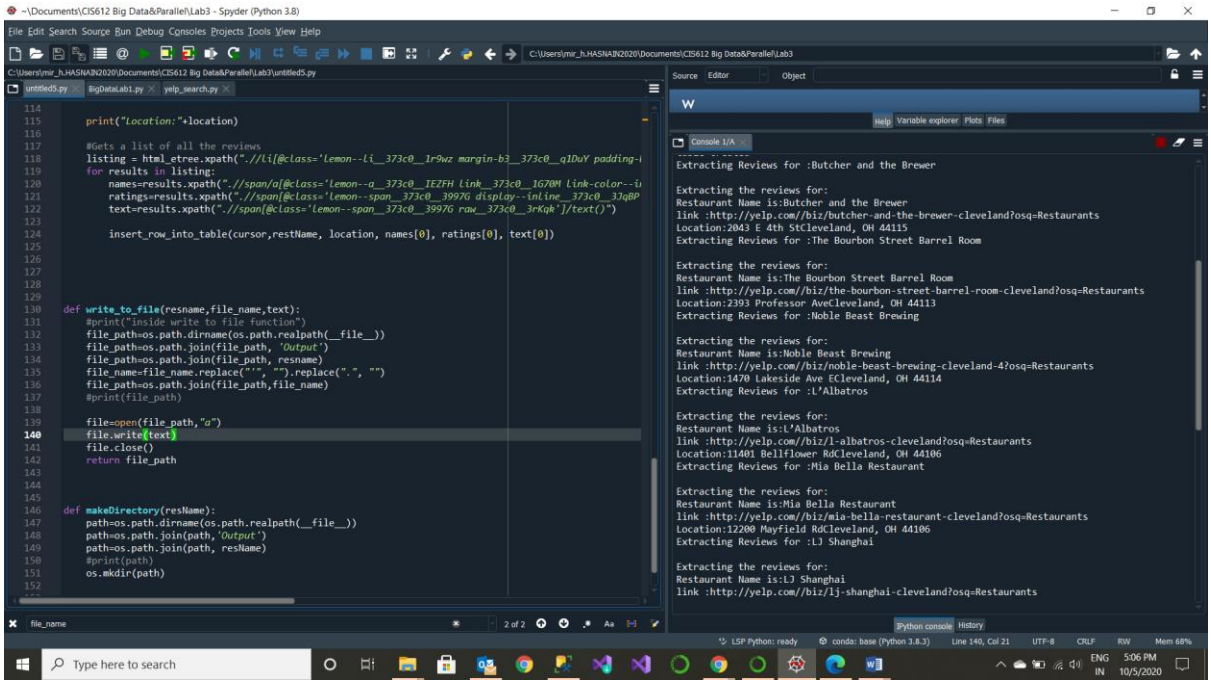
Then we fetch the links and names of top 10 restaurants in Cleveland from the Yelp site . We get the Xpath for each element by inspecting in chrome's Dev section.



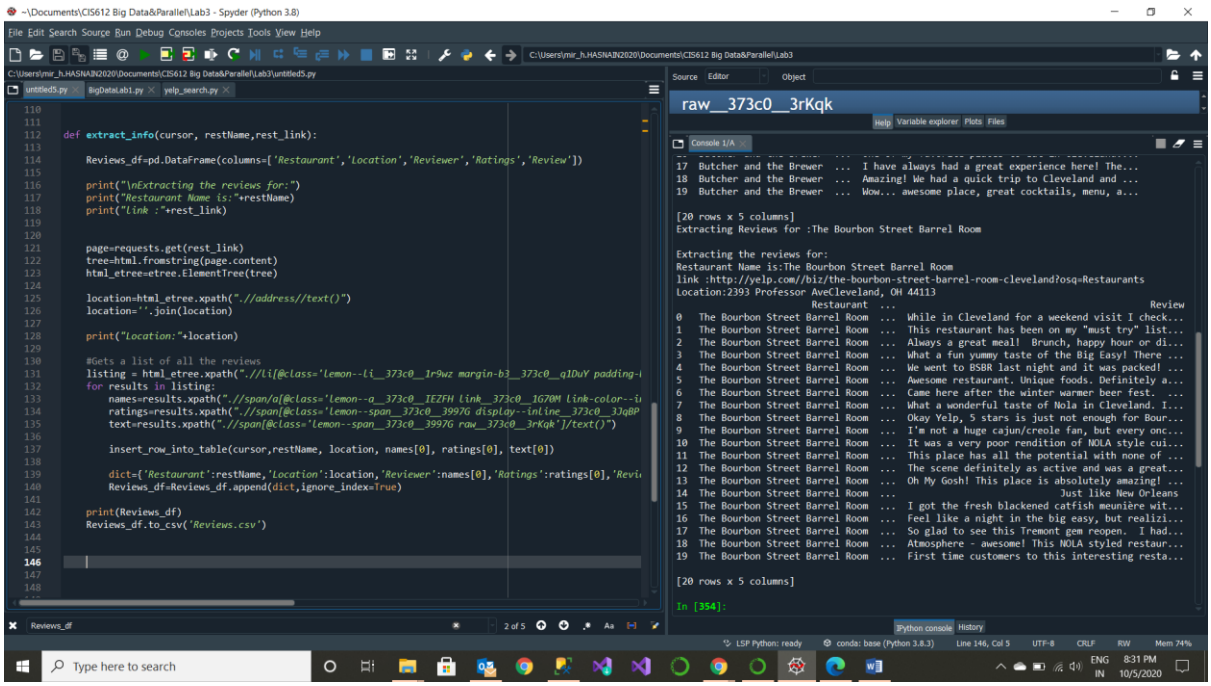


After getting the links of all restaurants we go to the each restaurants page and extract the reviews , location and rating .

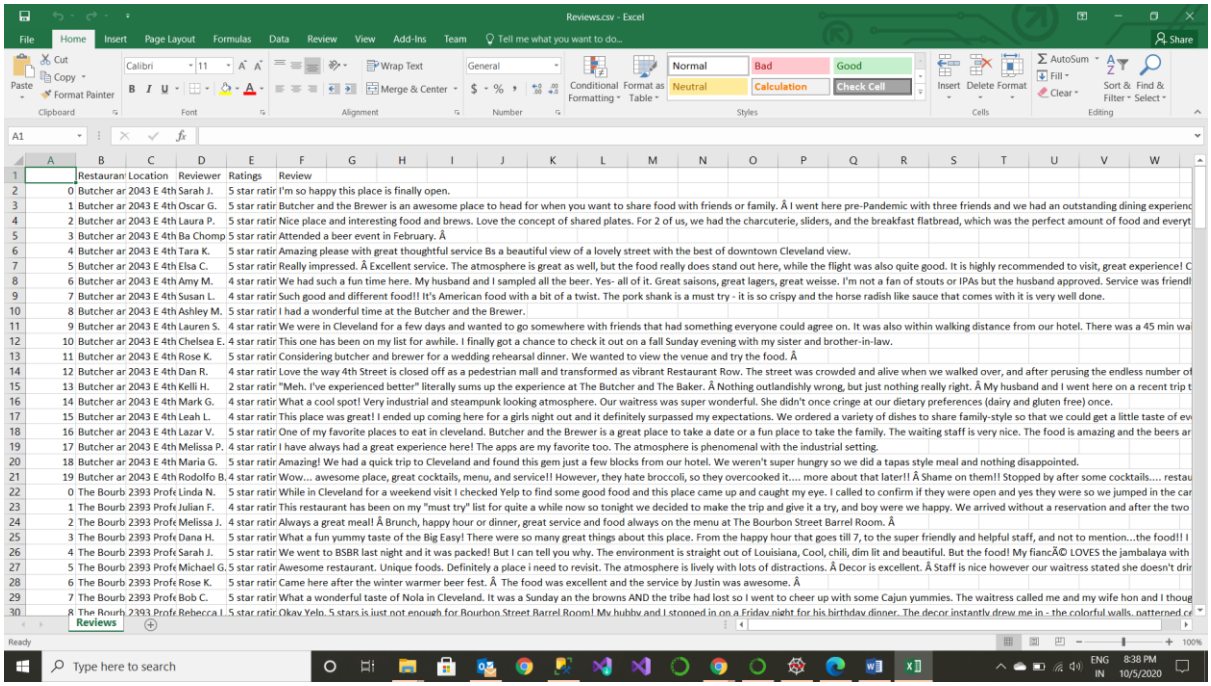




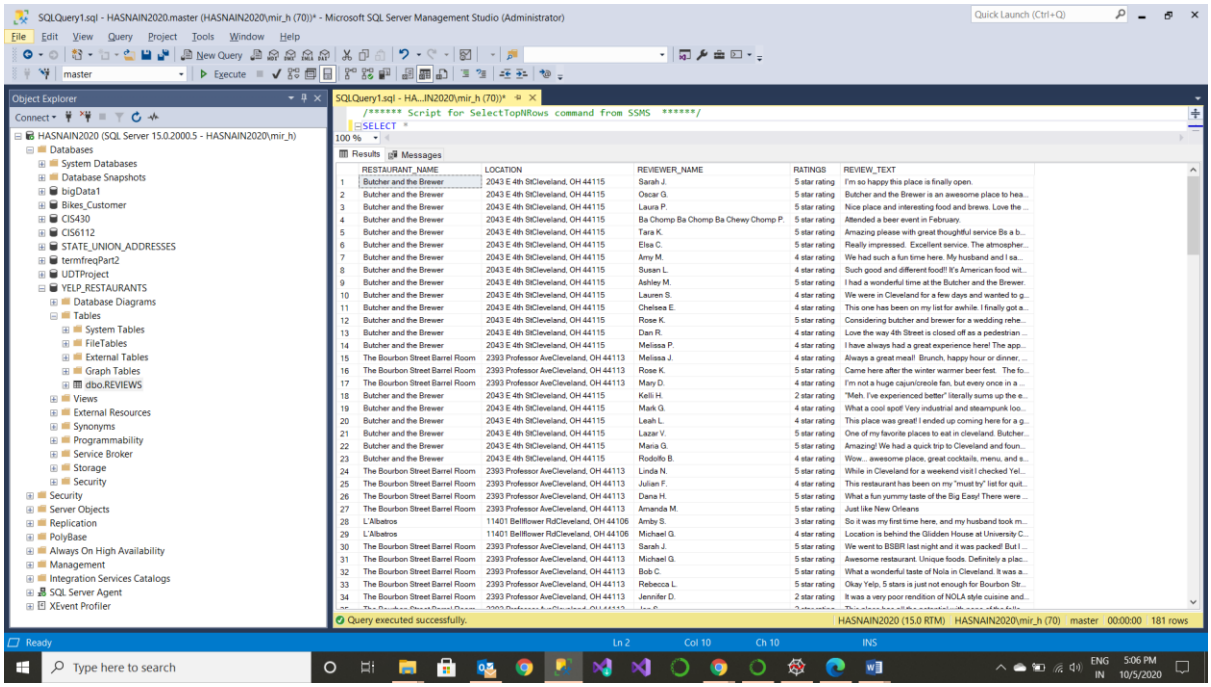
The extracted Reviews, ratings , text and Restaurant name are then inserted into the SQL database.



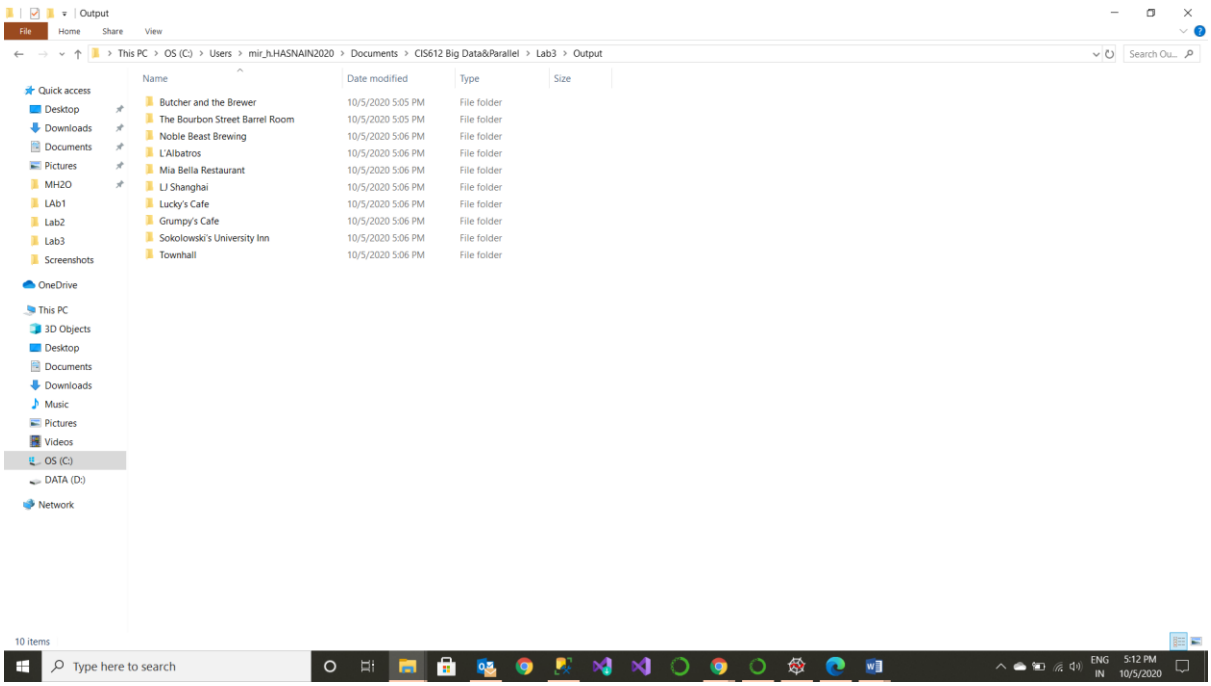
Put all the reviews into a csv file



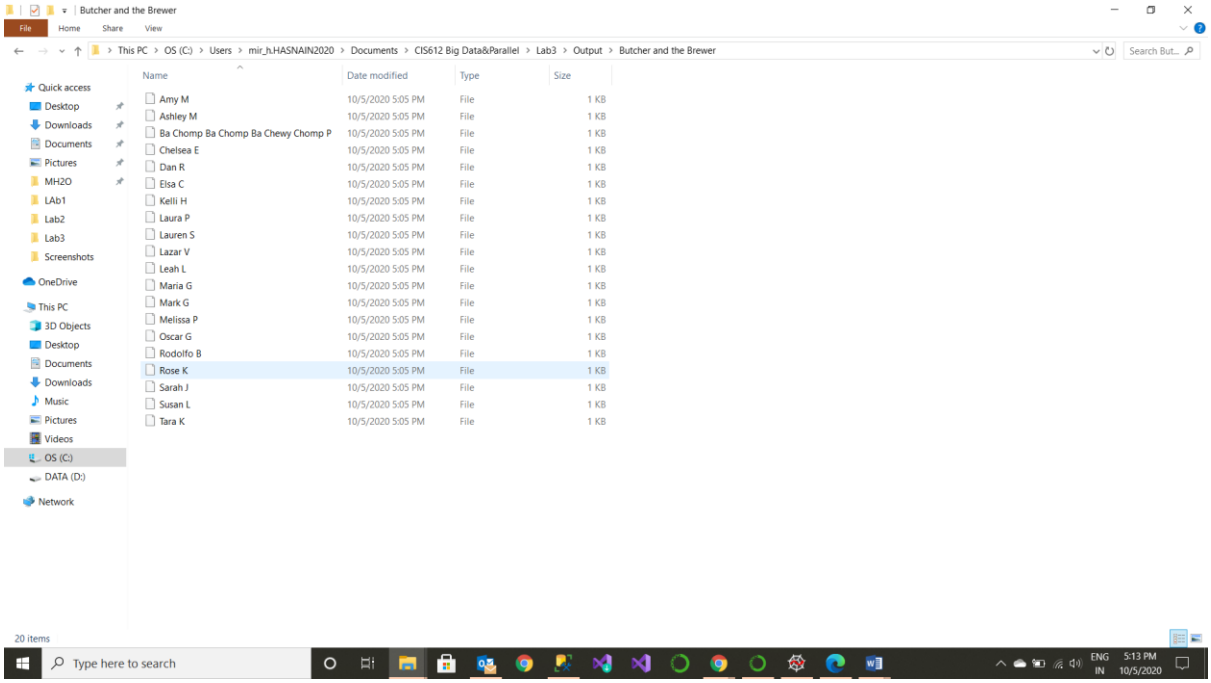
Also inserting the reviews into the SQL server



We also create folders of each restaurants in a Output folder



In each restaurant folder there are text files of the reviews of the restaurant.



Source Code:

```
from lxml import html,etree
from urllib.parse import urljoin
import requests
```

```
import os, pyodbc
import pandas as pd

url='https://www.yelp.com/search?find_desc=Restaurants&find_loc=Cle
veland%2C+OH'

server='HASNAIN2020'
database='YELP_RESTAURANTS'
table='REVIEWS'


def main():


   #load the yelp main page
   page=requests.get(url)
   tree=html.fromstring(page.content)
   html_etree=etree.ElementTree(tree)

   #getting the names and links of all restaurants
   print("getting a list of all the restaurants\n")
   links=html_etree.xpath(".//span[@class='lemon--
span__09f24__3997G text__09f24__2tZKC text-color--black-
regular__09f24__1QxyO text-align--left__09f24__3Drs0 text-weight--
bold__09f24__WGVdT text-size--inherit__09f24__2rwpp']/a")
   #putting them in a data frame and then later to a csv file
   Rest_df=pd.DataFrame(columns=['Name','link'])
   for link in links:
      Rest_df=Rest_df.append({'Name':link.attrib['name'],
'link':'http://yelp.com/'+link.attrib['href']},ignore_index=True)

   print(Rest_df)
   Rest_df.to_csv('links.csv')


   #creating a table and connecting to the server
   cursor=connect_to_sql_server()
```

```
    #fetching and putting the reviews for each restaurant

    for i in range(1,len(Rest_df)):
        print("Extracting Reviews for :"+Rest_df['Name'].values[i])
        makeDirectory(Rest_df['Name'].values[i])

extract_info(cursor,Rest_df['Name'].values[i],Rest_df['link'].values[i])

def connect_to_sql_server():
    #connect to sql server
    print("Connecting to the server")
    odbc_conn=pyodbc.connect('DRIVER={SQL
SERVER};SERVER='+server+';Trusted_Connection=yes;')
    odbc_conn.autocommit=True
    cursor=odbc_conn.cursor()

    #create db if does not exist
    transaction="IF DB_ID('{0}') IS  NULL CREATE DATABASE
{0};".format(database)
    cursor.execute(transaction)
    if(cursor==True):
        print("created  db")

    transaction="USE {0}".format(database)
    cursor.execute(transaction)
    if(cursor==True):
        print("USe  db")

    #drop table if exists
    transaction="IF OBJECT_ID('dbo.{0}') IS NOT  NULL DROP TABLE
dbo.{0};".format(table)
    cursor.execute(transaction)



    #create table
    print("table created")
```

```
    transaction=" CREATE TABLE dbo.{0} (RESTAURANT_NAME
VARCHAR(30),LOCATION VARCHAR(50), REVIEWER_NAME
VARCHAR(45),RATINGS VARCHAR(15), REVIEW_TEXT
NVARCHAR(MAX));".format(table)
    cursor.execute(transaction)

    return cursor

def insert_row_into_table(cursor,name,location,reviewer,ratings,text):
    text=text.replace("'","''")
    reviewer=reviewer.replace("'","''")
    name=name.replace("'","")


    #create databse if doesnt exist

    transaction="INSERT INTO {0} VALUES(
'{1}','{2}','{3}','{4}','{5}');".format(table,name,location,reviewer,ratings,te
xt)
    cursor.execute(transaction)
    write_to_file(name,reviewer,text)




def extract_info(cursor, restName,rest_link):




    print("\nExtracting the reviews for:")
    print("Restaurant Name is:"+restName)
    print("link :"+rest_link)
```

```
    page=requests.get(rest_link)
    tree=html.fromstring(page.content)
    html_etree=etree.ElementTree(tree)

    location=html_etree.xpath(".//address//text()")
    location=''.join(location)

    print("Location:"+location)

    #Gets a list of all the reviews
    listing = html_etree.xpath(".//li[@class='lemon--li__373c0__1r9wz
margin-b3__373c0__q1DuY padding-b3__373c0__342DA border--
bottom__373c0__3qNtD border-color--default__373c0__3-ifU']")
    for results in listing:
        names=results.xpath(".//span/a[@class='lemon--a__373c0__IEZFH
link__373c0__1G70M link-color--inherit__373c0__3dzpk link-size--
inherit__373c0__1VFlE']/text()")
        ratings=results.xpath(".//span[@class='lemon--
span__373c0__3997G display--inline__373c0__3JqBP border-color--
default__373c0__3-ifU']/div/@aria-label")
        text=results.xpath(".//span[@class='lemon--span__373c0__3997G
raw__373c0__3rKqk']/text()")

        insert_row_into_table(cursor,restName, location, names[0],
ratings[0], text[0])




def write_to_file(resname,file_name,text):
    #print("inside write to file function")
    file_path=os.path.dirname(os.path.realpath(__file__))
    file_path=os.path.join(file_path, 'Output')
    file_path=os.path.join(file_path, resname)
    file_name=file_name.replace("'", "").replace(".", "")
    file_path=os.path.join(file_path,file_name)
```

```
    #print(file_path)

    file=open(file_path,"a")
    file.write(text)
    file.close()
    return file_path



def makeDirectory(resName):
    path=os.path.dirname(os.path.realpath(__file__))
    path=os.path.join(path,'Output')
    path=os.path.join(path, resName)
    #print(path)
    os.mkdir(path)



main()
```