The main objective of this project is to web scrape an interactive map data from a web site to a data sets in excel file. With the help of Python, Python Libraries and packages the main objectives of scraping data from interactive map website was able to accomplish.

#The list of Python libraries used to connect the Python tool to the website are below

import pathlib  
from urllib.error import HTTPError  
from datetime import date, timedelta  
from collections import deque  
from bs4 import BeautifulSoup as soup # HTML data structure  
from urllib.request import urlopen as uReq # Web client  
import requests  
import re  
import json  
import ast  
from spinner import Spinner  
import pandas as pd

#Python Def function scripted in order to connect to website asking for request access. Using the HTML source code we found out the actual website to connect to the Def function scripted to the website.

#'https://ctitowers.com/wp-admin/admin-ajax.php’

def get\_html\_block(id\_str):  
 headers = {  
 'authority': 'ctitowers.com',  
 'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/84.0.4147.89 Safari/537.36',  
 'content-type': 'application/x-www-form-urlencoded; charset=UTF-8',  
 'accept': '\*/\*',  
 'origin': 'https://ctitowers.com',  
 'sec-fetch-site': 'same-origin',  
 'sec-fetch-mode': 'cors',  
 'sec-fetch-dest': 'empty',  
 'referer': 'https://ctitowers.com/cti-towers-site-locator/',  
 'accept-language': 'en-US,en;q=0.9',  
 }  
  
 data = {  
 'action': 'mmm\_async\_content\_marker',  
 'id': id\_str  
 }  
  
 response = requests.post('https://ctitowers.com/wp-admin/admin-ajax.php', headers=headers, data=data)  
  
 return response.text

#Once the connection is made to the website and the request is made to the website using above Def function script. Using beautiful soup python library the objective to scrape data and read ‘HTML Parser’. Once that done another Def Function is scripted in order to scrape the data in a presentable rows and columns.

page\_url = "https://ctitowers.com/cti-towers-site-locator/"  
uClient = uReq(page\_url)  
page\_soup = soup(uClient.read(), "html.parser")  
uClient.close()  
  
data = page\_soup.find\_all("script")[-2].string  
  
data = data.split('"markers":')[1].split('}];')[0].strip()  
maps\_list = ast.literal\_eval(data)  
  
  
  
  
  
  
def getField(html, field\_name):  
 # *TODO - add some error checking for when not found, etc.* field = html.split(field\_name + ":</b>")[1].split("<")[0].strip()  
 #print(field)  
 return field  
  
Record = []  
for item in maps\_list:  
 htmltxt = get\_html\_block(item.get('id'))  
 item\_soup = soup(htmltxt, 'lxml')  
 data = item\_soup.find\_all("li", class\_="adresse")  
 data2 = item\_soup.find\_all("h2")  
 ID\_Name = data2[0].text  
 Address = data[0].text  
 Lat\_Long = data[1].text  
 Tower\_Type = getField(htmltxt, "Tower Type")  
 STRUCTURE\_Height = getField(htmltxt, "Structure Height")  
 Ground\_Elevation = getField(htmltxt, "Ground Elevation")  
 County = getField(htmltxt, "County")  
 Account\_Manager = getField(htmltxt, "Account Manager")  
 Project\_Manager = getField(htmltxt, "Project Manager")  
 data3 = item\_soup.find\_all("li", class\_="telephone")  
 Telephone = data3[0].text  
 Record.append(  
 (ID\_Name, Address, Lat\_Long, Tower\_Type, STRUCTURE\_Height, Ground\_Elevation, County, Account\_Manager, Project\_Manager, Telephone)  
 )  
df = pd.DataFrame(Record)  
df.columns= ['ID\_Name', 'Address', 'Lat\_Long', 'Tower\_Type', 'STRUCTURE\_Height', 'Ground\_Elevation', 'County', 'Account\_Manager', 'Project\_Manager', 'Telephone']  
df\_CTI = df.drop\_duplicates(subset=['ID\_Name'], keep= 'last')  
print(df\_CTI)  
writer = pd.ExcelWriter('CTI\_Tower.xlsx')  
df\_CTI.to\_excel(writer, sheet\_name='Sheet1', index=False)  
writer.save()  
s.stop()