# **ITL LAB: EXPERIMENT 6**

**NAME: Shreya Shetty** 

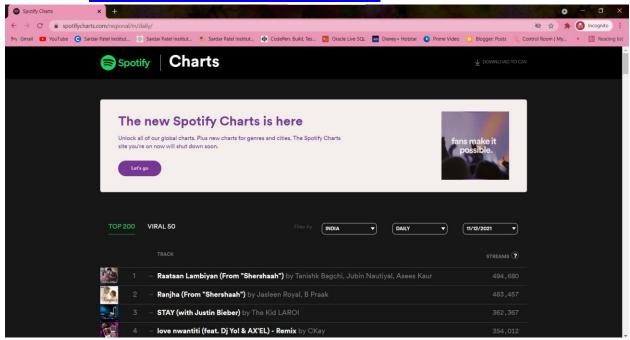
UID: 2019140059

**CLASS: TE IT** 

**BATCH: D** 

**AIM:** Write a python code to demonstrate the behaviour of Web Crawlers/ spiders (use XPATH, CSSPATH), extract information and store it in database.

WEBSITE: https://spotifycharts.com/regional/in/daily/



#### **URLs Crawled:**

https://spotifycharts.com/regional/in/daily/2021-10-1

https://spotifycharts.com/regional/in/daily/2021-10-2

.

https://spotifycharts.com/regional/in/daily/2021-11-1

#### **CODE:**

```
from bs4 import BeautifulSoup
import pandas as pd
import requests
from time import sleep
from datetime import date, timedelta
from lxml import etree
dir="D:\PROJECT AND CODES\ITL-Web-Scraper"
#create empty arrays for data we're collecting
dates=[]
url list=[]
final = []
#map site
headers = {
    "User-Agent":
    "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/72.0.3538.102 Safari/537.36
Edge/18.19582"
url = "https://spotifycharts.com/regional/in/daily/"
start date= date(2020, 11, 1)
end_date= date(2021, 11, 1)
delta= end date-start date
for i in range(delta.days+1):
    day = start_date+timedelta(days=i)
    day string= day.strftime("%Y-%m-%d")
    dates.append(day_string)
# Adding Urls to Scrape to url list
def add_url():
    for date in dates:
        c string = url+date
        url list.append(c string)
add_url()
# count stores the total number of songs scraped
count=0
# Function for going through each row in each url and finding
relevant song info
```

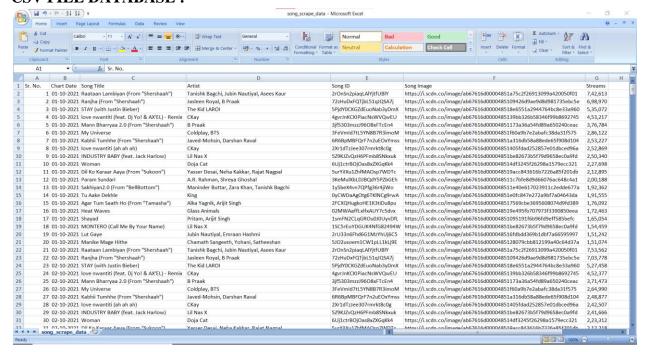
```
def song scrape(x):
    global count
    song count date=0
    pg = x
    for tr in songs.find("tbody").findAll("tr"):
        song count date+=1
        # Scraping data of top 20 songs on that particular date
'x' from url 'u'
        if song count date>20:
            break
        # Usig css selectors to scrape data with BeautifulSou[]
        artist= tr.find("td", class_="chart-table-
track").find("span").text
        artist= artist.replace("by ","").strip()
        title= tr.find("td", class = "chart-table-
track").find("strong").text
        songId= tr.find("td", {"class": "chart-table-
image"}).find("a").get("href")
        songId= songId.split("track/")[1]
        # Using XPath to scrape data with BeautifulSoup
xpath_id='//*[@id="content"]/div/div/div/span/table/tbody/tr['+st
r(song_count_date)+']/td[1]/a/img/@src'
        songImg=dom.xpath(xpath id)[0]
xpath_id='//*[@id="content"]/div/div/div/span/table/tbody/tr['+st
r(song count date)+']/td[5]/text()'
        songCount=dom.xpath(xpath id)[0]
        url date= x.split("daily/")[1]
        count+=1
        final.append([count, url date, title, artist, songId,
songImg, songCount])
        print("Top Song ", song count date," Data added")
# Loop through urls to create an array of all of our song info
for u in url list:
```

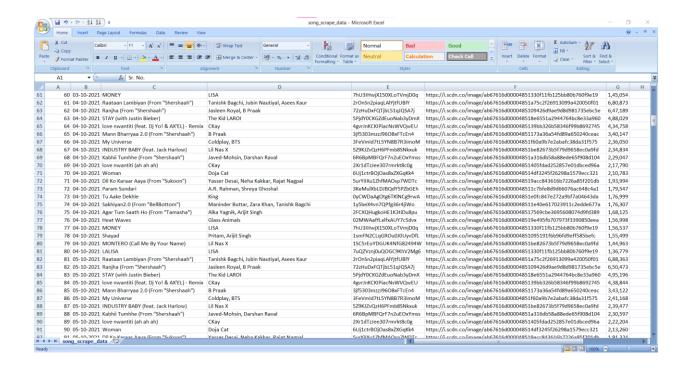
```
print('URL : ',u)
    read_pg= requests.get(u, headers=headers)
    # sleep(2)
    soup= BeautifulSoup(read pg.text, "html.parser")
    songs= soup.find(class = "chart-table")
    dom = etree.HTML(str(soup))
    song_scrape(u)
# Converting final data to data frame with pandas for easier data
manipulation
final_df = pd.DataFrame(final, columns= ["Sr. No.", "Chart Date",
"Song Title", "Artist", "Song ID", "Song Image", "Streams"])
print('Total count of songs is ',count)
print('Final Dataframe is created using pandas')
# Write data to csv
with open(dir+'\song_scrape_data.csv', 'w') as f:
    final_df.to_csv(f, header= True, index=False,
line terminator='\n')
   print('CSV file created successfully')
```

### **OUTPUT:**

```
Top Song
              Data added
Гор Song
              Data added
Top Song
             Data added
Top Song
             Data added
Top Song
              Data added
Top Song
              Data added
Top Song
              Data added
Top Song
              Data added
Top Song
              Data added
              Data added
Top Song
Top Song
              Data added
URL: https://spotifycharts.com/regional/in/daily/2021-11-01
Top Song 1 Data added
Top Song
              Data added
Top Song
              Data added
Top Song
              Data added
Top Song
              Data added
Top Song
              Data added
Top Song
              Data added
Top Song
              Data added
Top Song
              Data added
              Data added
Top Song
Top Song
              Data added
Top Song
          20 Data added
Total count of songs is 640
Final Dataframe is created using pandas
CSV file created successfully
PS D:\PROJECT AND CODES>
```

## **CSV FILE DATABASE:**





#### **CONCLUSION:**

Hence, I was successfully able to web crawl multiple URLS (<a href="https://spotifycharts.com/regional/in/daily/">https://spotifycharts.com/regional/in/daily/</a>) related to my case study using both CSS and XPath with BeautifulSoup in Python to extract and store data in CSV file of the Top 20 songs in Spotify Charts in last 1 month.