# Aim: Web Scrapping using Python

**IDE:** Tableau

# Theory:

Web scraping is a valuable technique for extracting data from websites for various purposes such as research, analysis, and automation. In this lab manual, we will explore the fundamentals of web scraping using Python, one of the most popular programming languages for this task. Through hands-on exercises and examples, students will learn how to retrieve data from web pages, parse HTML content, handle different types of data, and store the extracted information for further analysis.

Web scraping, web harvesting, or web data extraction is [data scraping](https://en.wikipedia.org/wiki/Data_scraping) used for [extracting](https://en.wikipedia.org/wiki/Data_extraction) [data](https://en.wikipedia.org/wiki/Data_extraction) from websites. Web scraping software may directly access the [World Wide Web](https://en.wikipedia.org/wiki/World_Wide_Web) using the [Hypertext](https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol) [Transfer Protocol](https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol) or a web browser. While web scraping can be done manually by a software user, the term typically refers to automated processes implemented using a [bot](https://en.wikipedia.org/wiki/Internet_bot) or [web crawler](https://en.wikipedia.org/wiki/Web_crawler). It is a form of copying in which specific data is gathered and copied from the web, typically into a central local [database](https://en.wikipedia.org/wiki/Database) or spreadsheet, for later [retrieval](https://en.wikipedia.org/wiki/Data_retrieval) or [analysis](https://en.wikipedia.org/wiki/Data_analysis).

Scraping a web page involves fetching it and extracting from it. Fetching is the downloading of a page (which a browser does when a user views a page). Therefore, web crawling is a main component of web scraping, to fetch pages for later processing. Once fetched, extraction can take place. The content of a page may be [parsed](https://en.wikipedia.org/wiki/Parsing), searched and reformatted, and its data copied into a spreadsheet or loaded into a database. Web scrapers typically take something out of a page, to make use of it for another purpose somewhere else. An example would be finding and copying names and telephone numbers, companies and their URLs, or e-mail addresses to a list (contact scraping).

As well as [contact scraping](https://en.wikipedia.org/wiki/Contact_scraping), web scraping is used as a component of applications used for [web indexing](https://en.wikipedia.org/wiki/Web_indexing), [web](https://en.wikipedia.org/wiki/Web_mining) [mining](https://en.wikipedia.org/wiki/Web_mining) and [data mining](https://en.wikipedia.org/wiki/Data_mining), online price change monitoring and [price comparison](https://en.wikipedia.org/wiki/Comparison_shopping_website), product review scraping (to watch the competition), gathering real estate listings, weather data monitoring, [website change detection](https://en.wikipedia.org/wiki/Change_detection_and_notification), research, tracking online presence and reputation, [web mashup](https://en.wikipedia.org/wiki/Web_mashup), and [web data integration](https://en.wikipedia.org/wiki/Web_data_integration). [Web pages](https://en.wikipedia.org/wiki/Web_page) are built using text-based mark-up languages ([HTML](https://en.wikipedia.org/wiki/HTML) and [XHTML](https://en.wikipedia.org/wiki/XHTML)), and frequently contain a wealth of useful data in text form. However, most web pages are designed for human [end-users](https://en.wikipedia.org/wiki/End-user_(computer_science)) and not for ease of automated use. As a result, specialized tools and software have been developed to facilitate the scraping of web pages.

Newer forms of web scraping involve monitoring data feeds from web servers. For example, [JSON](https://en.wikipedia.org/wiki/JSON) is commonly used as a transport mechanism between the client and the web server. There are methods that some websites use to prevent web scraping, such as detecting and disallowing bots from crawling (viewing) their pages. In response, there are web scraping systems that rely on using techniques in [DOM](https://en.wikipedia.org/wiki/Document_Object_Model) parsing, [computer](https://en.wikipedia.org/wiki/Computer_vision) [vision](https://en.wikipedia.org/wiki/Computer_vision) and [natural language processing](https://en.wikipedia.org/wiki/Natural_language_processing) to simulate human browsing to enable gathering web page content for offline parsing

# Pre Lab Exercise:

1. How do you retrieve HTML content from a web page using Python?
2. What are the steps involved in storing and saving scraped data for further analysis?
3. What are the potential challenges and limitations of web scraping?

# Tasks:

Perform the following tasks:

Task 1: Extract Table-1 from the web link <https://en.wikipedia.org/wiki/List_of_largest_companies_in_the_United_States_by_revenue>.

**Code :-**

import pandas as pd

import requests

from bs4 import BeautifulSoup

url = "https://en.wikipedia.org/wiki/List\_of\_largest\_companies\_in\_the\_United\_States\_by\_revenue"

Page = requests.get(url)

Soup = BeautifulSoup(Page.text, "html")

Table = Soup.find\_all("table")

Table = Table[1]

Headers = Table.find\_all("th")

Titles = [titles.text.strip() for titles in Headers]

Dataset = pd.DataFrame(columns=Titles)

Rows = Table.find\_all("tr")

for row in Rows[1:]:

Data = row.find\_all("td")

Row\_Data = [row.text.strip() for row in Data]

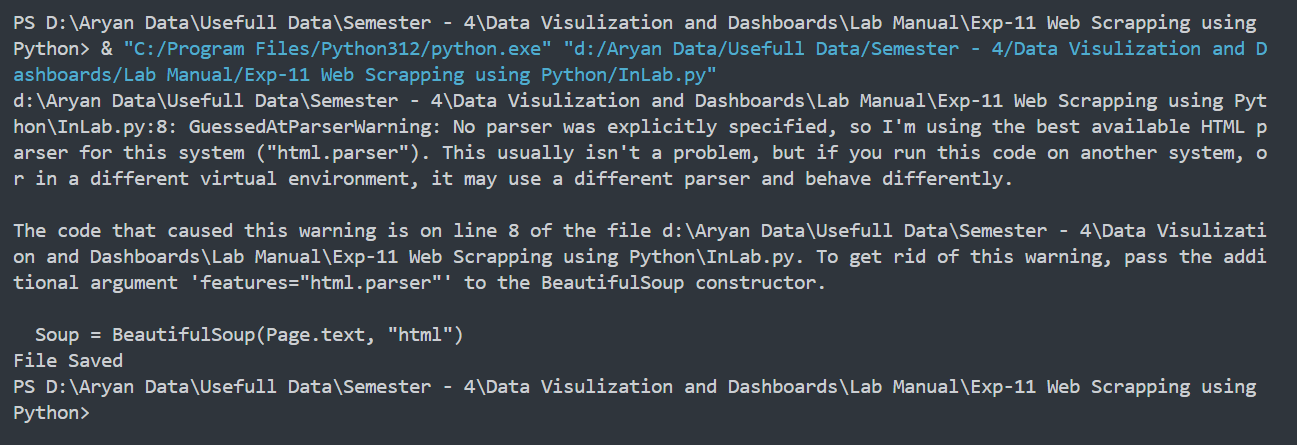
length = len(Dataset)

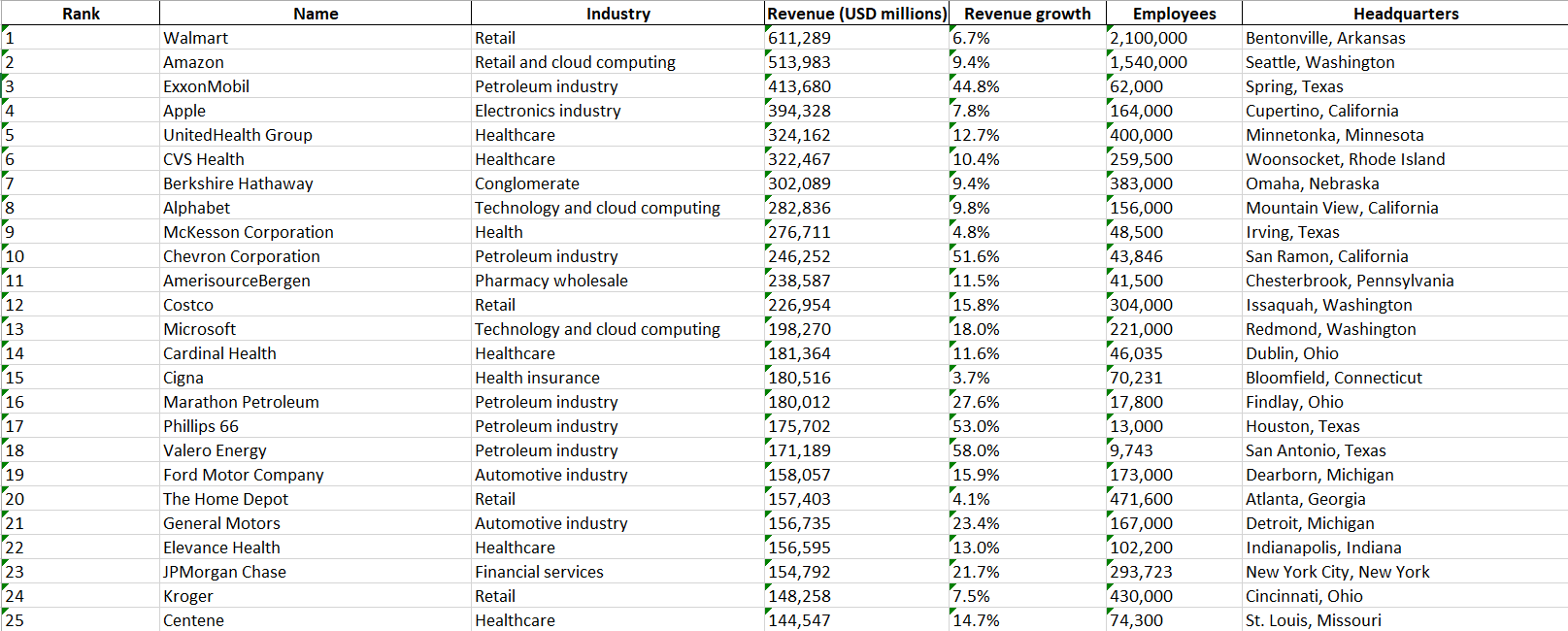
Dataset.loc[length] = Row\_Data

Dataset.to\_excel("D:/Aryan Data/Usefull Data/Semester - 4/Data Visulization and Dashboards/Lab/Web Scrapping/Largest Companies In USA by Revenue.xlsx",index=False,)

print("File Saved")

# Results:-





Task 2: Perform data analysis over:

1. Analyze the market share of different companies based on their revenue.

**Code:-**

Dataset["Revenue (USD millions)"] = (

Dataset["Revenue (USD millions)"].str.replace(",", "").astype(float))

total\_revenue = Dataset["Revenue (USD millions)"].sum()

Market\_Share = {"Company\_Name": [], "Share": []}

for index, row in Dataset.iterrows():

company\_name = row["Name"]

revenue = row["Revenue (USD millions)"]

share\_percentage = (revenue / total\_revenue) \* 100

Market\_Share["Company\_Name"].append(company\_name)

Market\_Share["Share"].append(share\_percentage)

Market\_Share = pd.DataFrame(Market\_Share)

Market\_Share["Share"] = Market\_Share["Share"].map("{:.2f}%".format)

print(Market\_Share)

# Results:-

1. Segment companies based on industry to analyze sectors separately.

**Code:-**

Industry = Dataset.groupby("Industry")["Name"].unique()

print(Industry)

# Results:-

1. Segment companis based on headquarters to analyze sectors separately.

**Code:-**

Headquarters = Dataset.groupby("Headquarters")["Name"].unique()

print(Headquarters)

# Results:-

1. Observe the growth of the company based on revenue vs employee size.

**Code:-**

revenue\_employee\_df = Dataset[['Revenue (USD millions)', 'Employees']]

plt.figure(figsize=(10, 6))

sns.scatterplot(x='Revenue (USD millions)', y='Employees', data=revenue\_employee\_df, s=100, color='blue', alpha=0.7)

plt.title('Revenue vs Employee Size')

plt.xlabel('Revenue (USD millions)')

plt.ylabel('Employees')

plt.grid(True)

sns.regplot(x='Revenue (USD millions)', y='Employees', data=revenue\_employee\_df, scatter=False, color='red')

plt.show()

# Results:-

# 

# Observation and Result Analysis:-

# Post Lab Exercise:

**Exercise-1:** Extract Table-1 from the weblink <https://en.wikipedia.org/wiki/List_of_largest_companies_in_India>

**Code:-**

import pandas as pd

import requests

from bs4 import BeautifulSoup

url = "https://en.wikipedia.org/wiki/List\_of\_largest\_companies\_in\_India"

Page = requests.get(url)

Soup = BeautifulSoup(Page.text, "html")

Table = Soup.find\_all("table")

Table = Table[0]

Headers = Table.find\_all("th")

Titles = [titles.text.strip() for titles in Headers]

Dataset = pd.DataFrame(columns=Titles)

Rows = Table.find\_all("tr")

for row in Rows[1:]:

Data = row.find\_all("td")

Data\_filtered = [Data[i] for i in range(len(Data)) if i not in [1, 3]]

Row\_Data = [row.text.strip() for row in Data\_filtered]

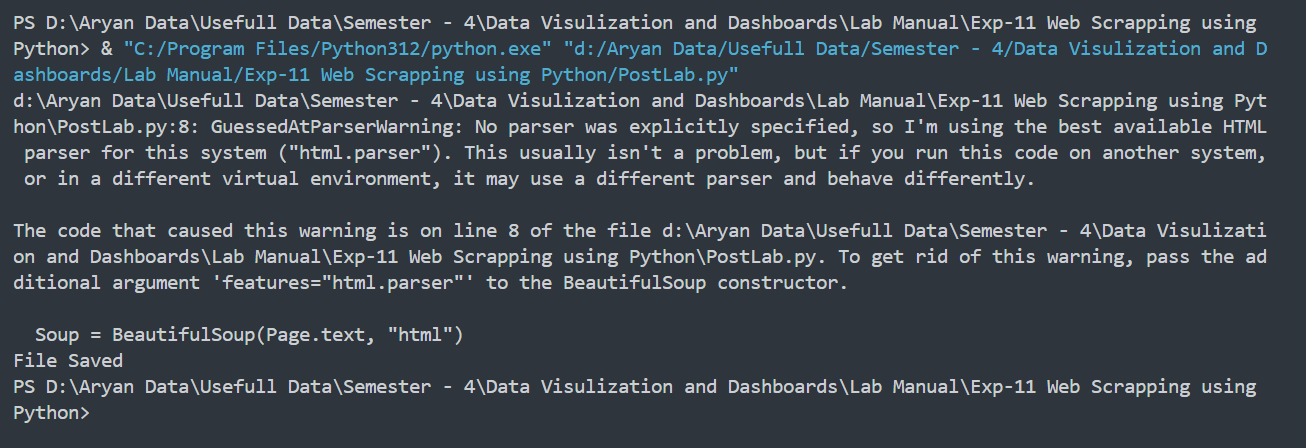
Dataset.loc[len(Dataset)] = Row\_Data

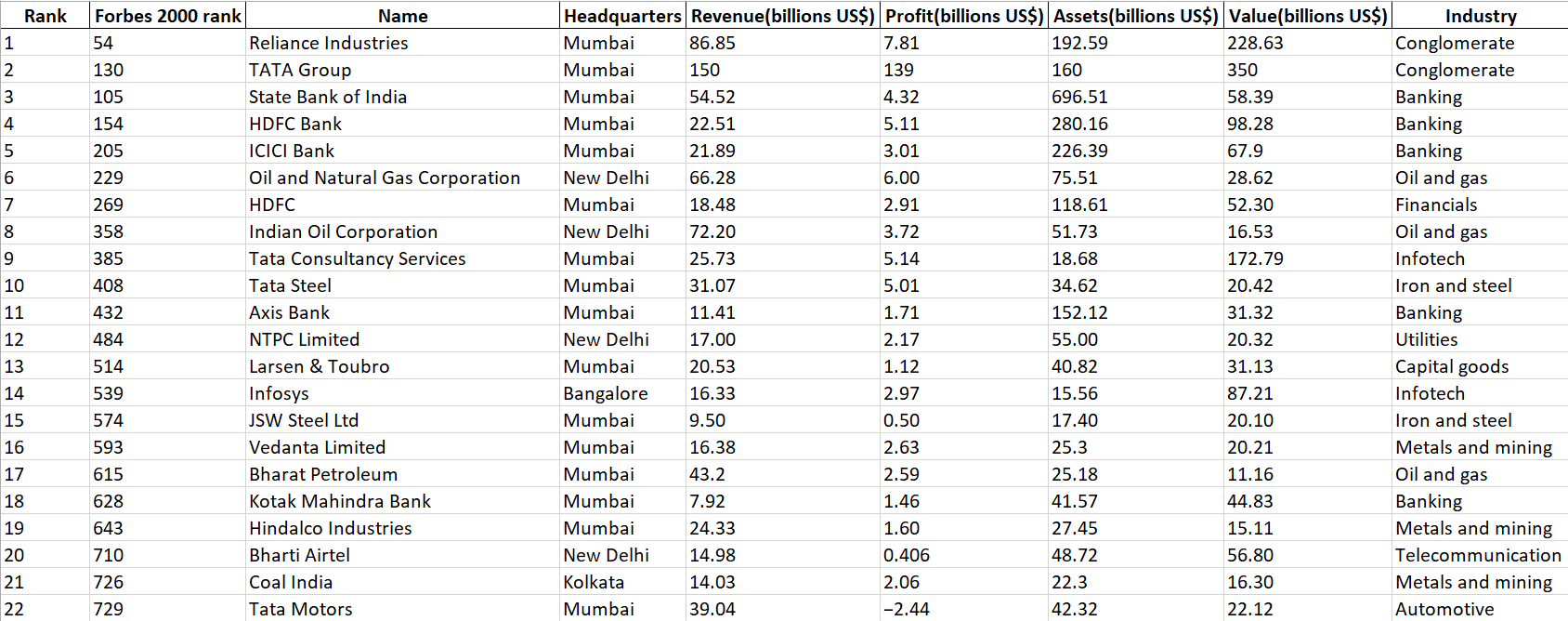
Dataset.to\_excel("D:/Aryan Data/Usefull Data/Semester - 4/Data Visulization and Dashboards/Lab Manual/Exp-11 Web Scrapping using Python/Largest Companies In India.xlsx",index=False,)

print("File Saved")

# 

# Results:-





Task 2: Perform data analysis over:

1. Analyze the market share of different companies based on their revenue.

**Code :-**

Dataset["Revenue(billions US$)"] = Dataset["Revenue(billions US$)"].astype(str)

Dataset["Revenue(billions US$)"] = (Dataset["Revenue(billions US$)"].str.replace(",", "").astype(float))

total\_revenue = Dataset["Revenue(billions US$)"].sum()

Market\_Share = {"Company\_Name": [], "Share": []}

for index, row in Dataset.iterrows():

company\_name = row["Name"]

revenue = row["Revenue(billions US$)"]

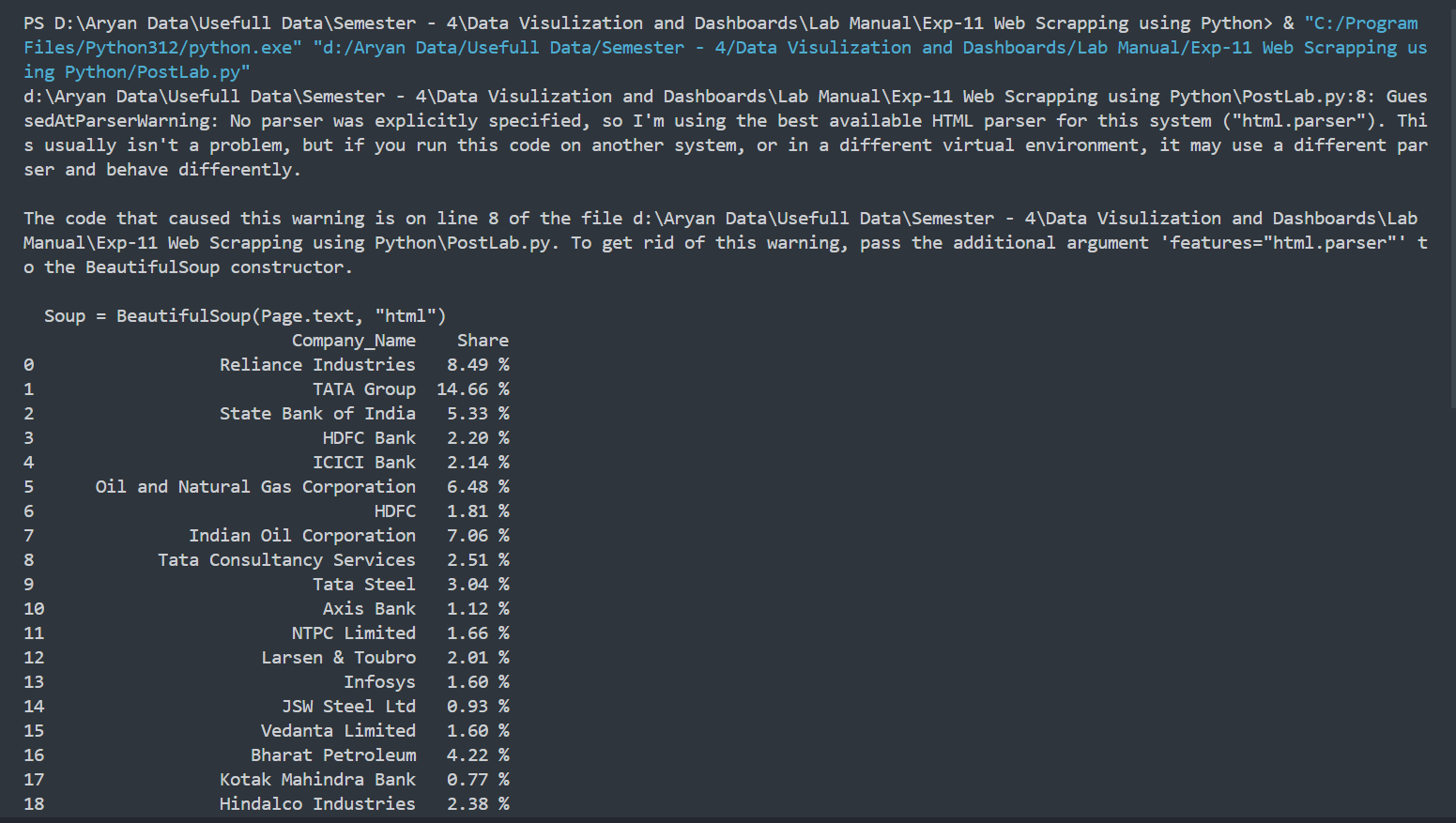
share\_percentage = (revenue / total\_revenue) \* 100

Market\_Share["Company\_Name"].append(company\_name)

Market\_Share["Share"].append(f"{share\_percentage:.2f} %")

Market\_Share = pd.DataFrame(Market\_Share)

print(Market\_Share)

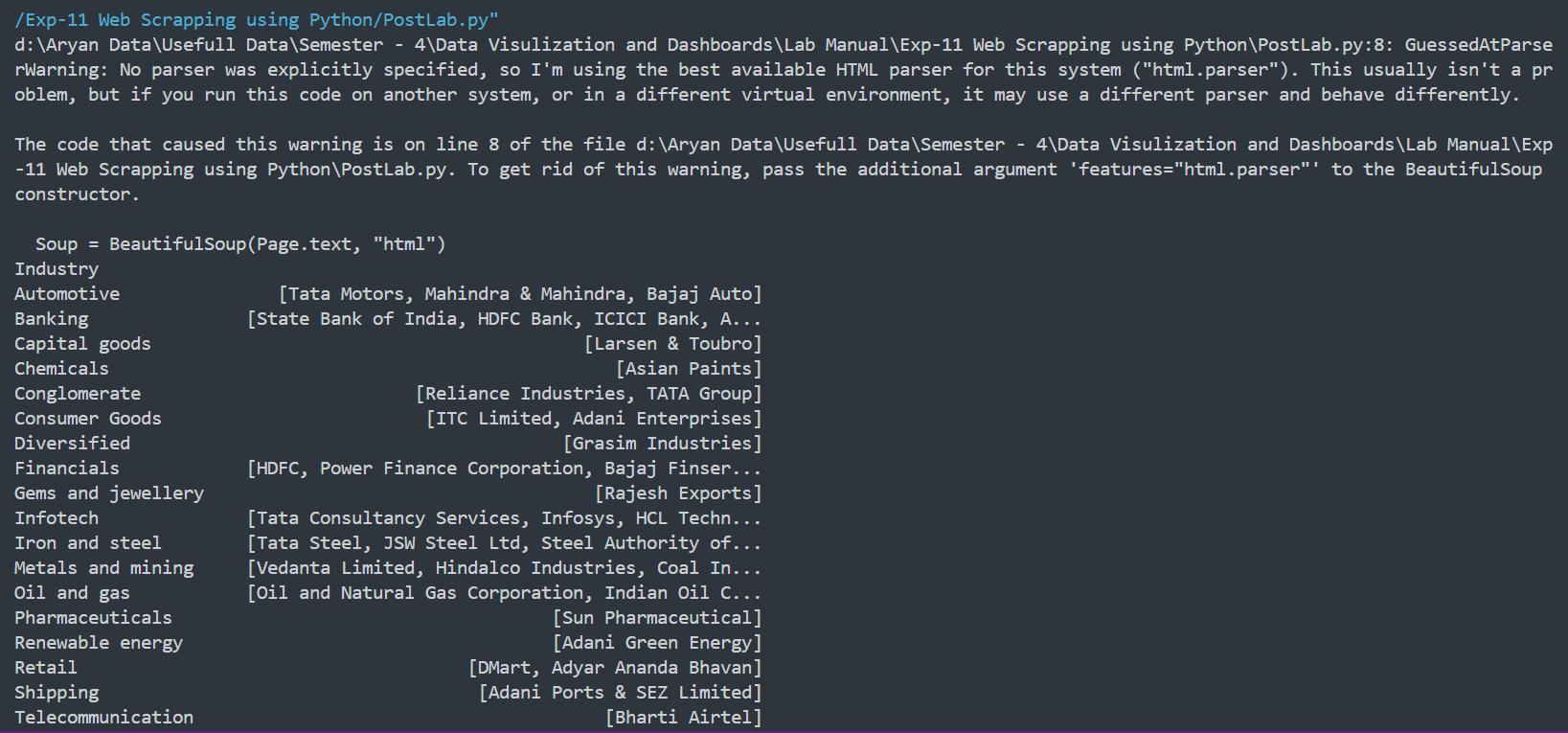
**Result:-**

1. Segment companies based on industry to analyze industry sectors separately.

**Code :-**

Industry = Dataset.groupby("Industry")["Name"].unique()

print(Industry)

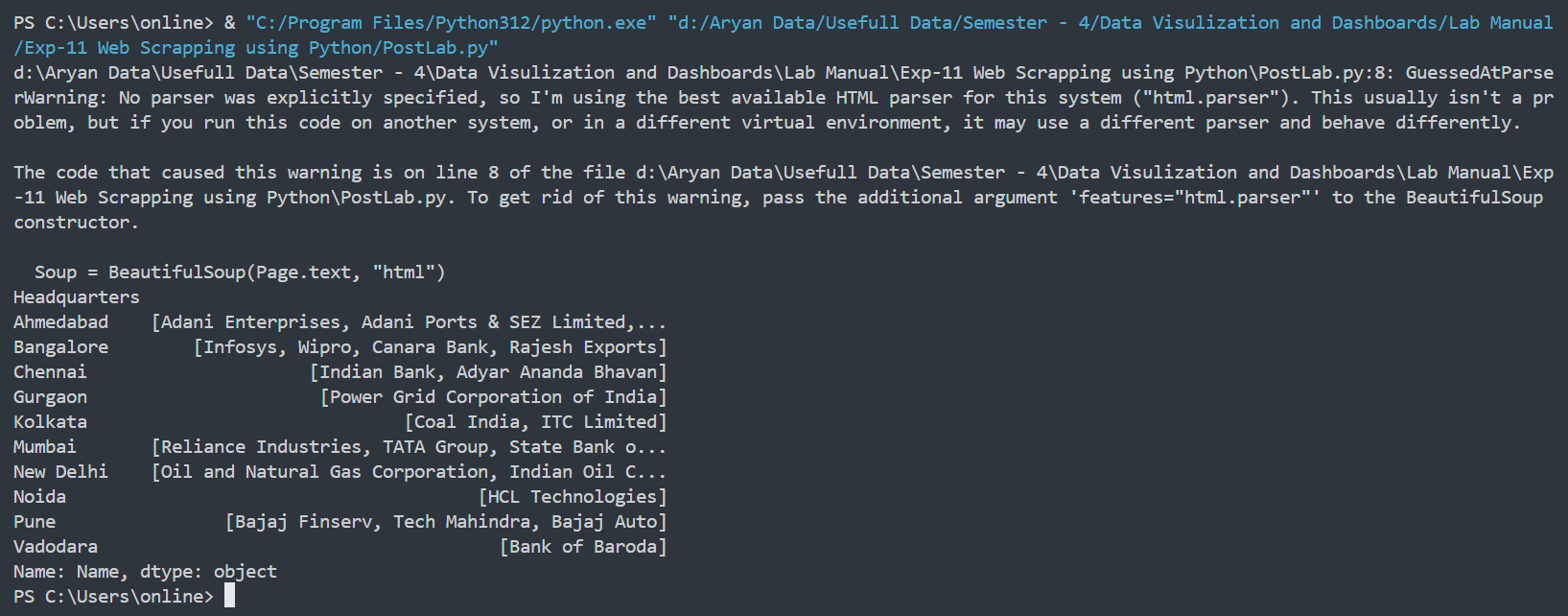
**Result :-**

1. Segment companies based on headquarters to analyze sectors separately.

**Code :-**

Headquarters = Dataset.groupby("Headquarters")["Name"].unique()

print(Headquarters)

**Result :-**