Topic	Web Scraping 2	
Class Description	**Students would be reworking the previously written code to scrape more data.**	
Class	**C128**	
Class time	**45 mins**	
Goal	• Scrape more data about all the exoplanets	

Resources Required	• Teacher Resources
	  ○ Laptop with internet connectivity
	  ○ Earphones with mic
	  ○ Notebook and pen
	• Student Resources
	  ○ Laptop with internet connectivity
	  ○ Earphones with mic
	  ○ Notebook and pen	

Class structure	**Warm Up** **Teacher-led Activity** **Student-led Activity** **Wrap up**	**5 mins** **15 min** **15 min** **5 min**

CONTEXT

- **Review the concepts learned in the earlier classes**

Class Steps	Teacher Action	Student Action
Step 1: Warm Up (5 mins)	Hi <Student Name>! In the last class, we scraped exoplanet's data from NASA's website. Can you recall all the tools that we used in the last class?	**ESR:** - Selenium - BeautifulSoup

	Great! Now, in today's class, we will scrape some more data from the same website. We got some data like distance from earth, planet size, etc. but today we will scrape more data so that when we perform analysis later, we can better predict the planets, for instance, to see if they are likely habitable, etc. Are you excited?	**ESR:** "Yes!"
	Before we start I have an exciting quiz question for you! Are you ready to answer this question? Teacher click on the  button on the bottom right corner of your screen to start the In-Class Quiz. A quiz will be visible to both you and the student. Encourage the student to answer the quiz question. The student may choose the wrong option, help the student to think correctly about the question and then answer again. After the student selects the correct option, the  button will start appearing on your screen. Click the End quiz to close the quiz pop-up and continue the class.	ESR: Yes!

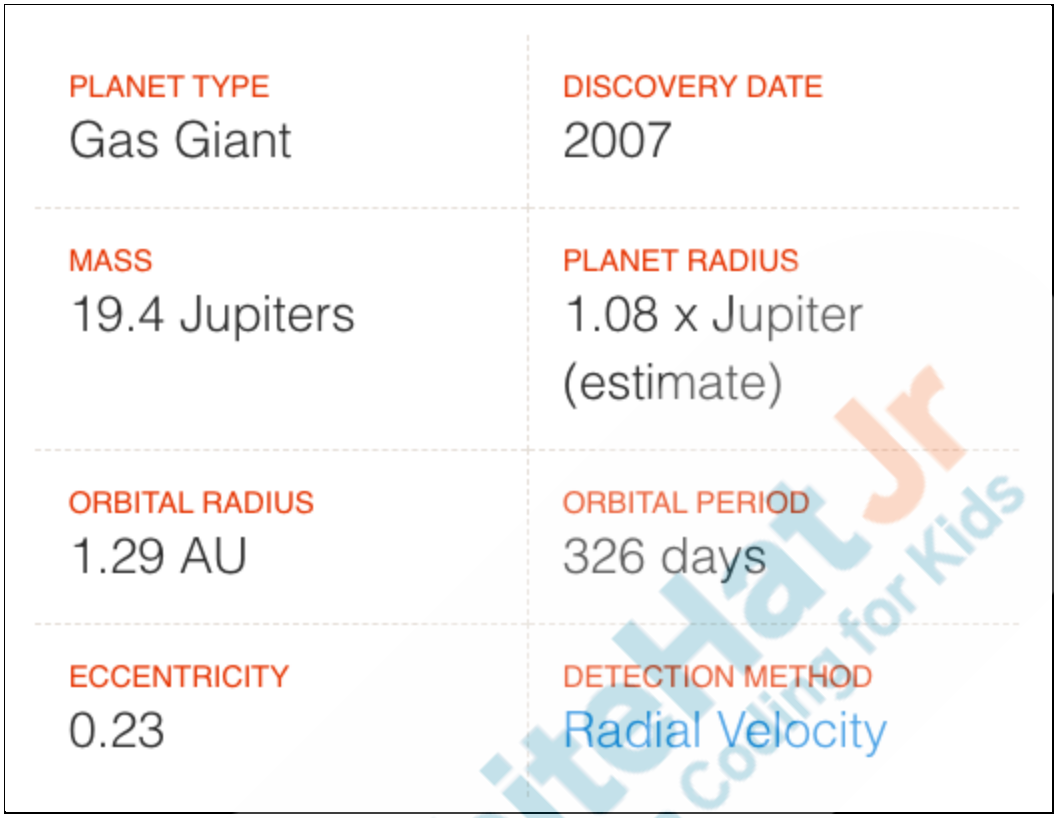
	Let's get started!	
Teacher Initiates Screen Share		
CHALLENGE • **Scraping more data from the website and letting students lead the development this time.**		
Step 2: Teacher-led Activity (15 min)	*Teacher opens the same website that we scraped in the last class.* *<Teacher opens the link from Teacher activity 1>* https://exoplanets.nasa.gov/exoplanet-catalog/	
	Let's look at this page again. Here, if we look closely, we can see that the name of these exo-planets is a hyperlink. ***Note**: NASA's exoplanet catalog web page keeps updating as per the new planet discoveries. At the time of writing this document, the web page had **428** with **10 Planets per page** showing a total of **planets data** 4280.* ***By default it can have 25 planets per page.***	

1 of 428 Per page 10

NAME ↑	LIGHT-YEARS FROM EARTH	PLANET MASS	STELLAR MAGNITUDE	DISCOVERY DATE
11 Comae Berenices b	305	19.4 Jupiters	4.74	2007
11 Ursae Minoris b	410	14.74 Jupiters	5.016	2009
14 Andromedae b	247	4.8 Jupiters	5.227	2008
14 Herculis b	59	4.66 Jupiters	6.61	2002
16 Cygni B b	69	1.78 Jupiters	6.25	1996
18 Delphini b	249	10.3 Jupiters	5.506	2008
1RXS J160929.1-210524 b	473	8 Jupiters	12.057	2008
24 Bootis b	314	0.91 Jupiters	5.58	2018
24 Sextantis b	236	1.99 Jupiters	6.441	2010
24 Sextantis c	236	0.86 Jupiters	6.441	2010

1 of 428 Back to top

Let's click on the link and see what kind of data we can find?



	Great! Now, let's say we want to scrape this data as well. Can you tell me what's the first change that we'll have to make in our previous code?	**ESR:** We need to save the hyperlink's href in our CSV.
	That's great! Let's get started. We will add a new column in our header. Our header variable would now look like this: `headers = ["name", "light_years_from_earth", "planet_mass", "stellar_magnitude", "discovery_date", "hyperlink"]`	

updated_scraper.py > scrape

```python
from selenium import webdriver
from selenium.webdriver.common.by import By
from bs4 import BeautifulSoup
import time
import csv

# NASA Exoplanet URL
START_URL = "https://exoplanets.nasa.gov/exoplanet-catalog/"

# Webdriver
browser = webdriver.Edge("C:/Whitehat_jr/PRO-127-130/msedgedriver.exe")
browser.get(START_URL)

time.sleep(10)

planets_data = []

headers = ["name", "light_years_from_earth", "planet_mass", "stellar_magnitude", "discovery_date", "hyperlink"]

```

We have added an extra hyperlink into our header list. Now, we also need to add this into the temp_list variable list, before we append into the planets_data.

Before we do that, let's investigate the href url in these hyperlinks:

Here, we can see that these links do not have https://exoplanets.nasa.gov before them. We will have to add them.

Now to achieve this, we will do the following:

Note: *Scrape 4 pages only. Thus in the for loop range is given from 1 to 5. If each page has data of 25 planets, total 100 hyperlinks will be scraped in the later section.*

```
def scrape():
    for i in range(1,5):
        while True:
            time.sleep(2)

            soup = BeautifulSoup(browser.page_source, "html.parser")


        for ul_tag in soup.find_all("ul", attrs={"class", "exoplanet"}):
            li_tags = ul_tag.find_all("li")
            temp_list = []
            for index, li_tag in enumerate(li_tags):
                if index == 0:
                    temp_list.append(li_tag.find_all("a")[0].contents[0])
                else:
                    try:
                        temp_list.append(li_tag.contents[0])
                    except:
                        temp_list.append("")

            # Get Hyperlink Tag
            hyperlink_li_tag = li_tags[0]

            temp_list.append("https://exoplanets.nasa.gov"+ hyperlink_li_tag.find_all("a", href=True)[0]["href"])

            planets_data.append(temp_list)

        browser.find_element(By.XPATH, value='//*[@id="primary_column"]/footer/div/div/div/nav/span[2]/a').click()

        print(f"Page {i} scraping completed")

# Calling Method
scrape()
```

We have added:

```
hyperlink_li_tag = li_tags[0]

temp_list.append("https://exoplanets.
nasa.gov"+hyperlink_li_tag.find_all("
a", href=True)[0]["href"])
```

Here, first we are creating a variable hyperlink_li_tag and then we are using this variable to find all the anchor tag with href, take the first anchor tag (since we know there's only one anchor tag in all li tags) and then we are taking out the href from it.

	Now that we have the links in planet_data, can you tell me what should be our next steps? Perfect, we will create a new function that will take these hyperlinks one by one, get the HTML and then we will scrape the data.	**ESR:** - We'll scrape data by using these links!
	To make sure that we are scraping pages one by one, we'll add a code in the **scrape()** function to check the current page number.	

```
def scrape():
    for i in range(1,5):
        while True:
            time.sleep(2)

            soup = BeautifulSoup(browser.page_source, "html.parser")

            # Check page number
            current_page_num = int(soup.find_all("input", attrs={"class", "page_num"})[0].get("value"))

            if current_page_num < i:
                browser.find_element(By.XPATH, value='//*[@id="primary_column"]/footer/div/div/div/nav/span[2]/a').click()
            elif current_page_num > i:
                browser.find_element(By.XPATH, value='//*[@id="primary_column"]/footer/div/div/div/nav/span[1]/a').click()
            else:
                break
```

	Earlier, we used selenium because we wanted to click a button on the page (next button) but this time, we do not want to interact with the browser, therefore we can do this without selenium. Let's get started!	
Teacher Stops Screen Share		
	Now it's your turn. Please share your screen with me.	

- **Ask Student to press ESC key to come back to panel**
- **Guide Student to start Screen Share**
- **Teacher gets into Fullscreen**

ACTIVITY

- **Student creates a new function to use all the hyperlinks one by one and scrape data from there**

Step 3: Student-Led Activity (15 min)	*Ask the student to move the variables **headers** and **planets_data** to the global scope, i.e, below time.sleep(10) line.* This is because we now would want to access these variables in multiple functions. Let's add the new headers, that is, the new data that is available on the new page we just discovered.	*The student moves the variables.*

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from bs4 import BeautifulSoup
import time
import pandas as pd
import requests
import csv

# NASA Exoplanet URL
START_URL = "https://exoplanets.nasa.gov/exoplanet-catalog/"

# Webdriver
browser = webdriver.Edge("C:/Whitehat_jr/PRO-127-130/msedgedriver.exe")
browser.get(START_URL)

time.sleep(10)

planets_data = []

headers = ["name", "light_years_from_earth", "planet_mass", "stellar_magnitude", "discovery_date", "hyperlink",
           "planet_type", "planet_radius", "orbital_radius", "orbital_period", "eccentricity"]

```

<table>
<tr>
<td></td>
<td>

```
headers = ["name",
"light_years_from_earth",
"planet_mass", "stellar_magnitude",
"discovery_date", "hyperlink",
"planet_type", "planet_radius",
"orbital_radius", "orbital_period",
"eccentricity"]
```

</td>
<td>The student adds more headers.</td>
</tr>
<tr>
<td></td>
<td>Great, now let's create a new function and call that function. We will call the function in loop and pass the hyperlink we saved with the earlier function into this function.

Also, let's comment out the CSV saving code. We want to save a csv with half the data, right?</td>
<td>The student creates a new function.</td>
</tr>
<tr>
<td colspan="3">

```
31      # with open("scrapper_2.csv", "w") as f:
32      #     csvwriter = csv.writer(f)
33      #     csvwriter.writerow(headers)
34      #     csvwriter.writerows(planet_data)
35
36  def scrape_more_data(hyperlink):
37      pass
38
39  scrape()
40  for data in planet_data:
41      scrape_more_data(data[5])
```

</td>
</tr>
</table>

	Okay, now earlier, we created a soup object where we passed the browser's page source and parsed it as html. This time, since we are not going to use selenium, how can we do it?	**ESR:** We can get the page's HTML by making a GET request.

	That's right! For that, we will import requests module
	
	`import requests`
	
	And we will write the following code inside the new function we created:
	
	`page = requests.get(hyperlink)`
	`soup = BeautifulSoup(page.content, "html.parser")`
	
	Here, we are first getting the page, and then we are parsing the contents of the page as HTML.
		The student follows instructions.

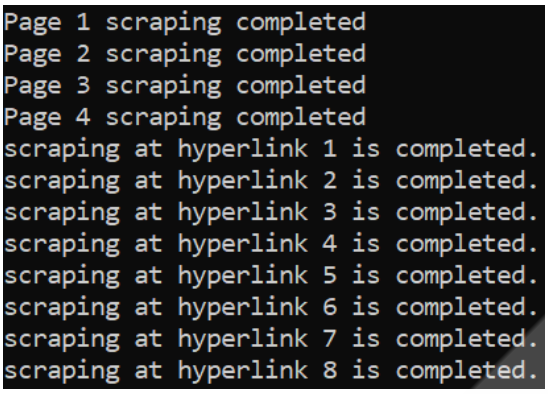
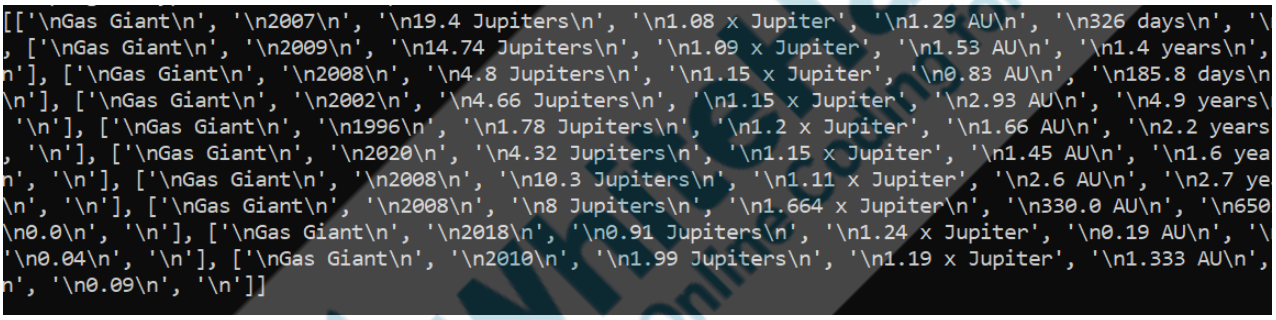
	Ask the student to create a new list ***new_planets_data*** *to save data from these new pages, and* ***ask them to scrape the data like before.*** *Help the student if required.* *The code* ***should look something like this:***	

```
new_planets_data = []

def scrape_more_data(hyperlink):
    try:
        page = requests.get(hyperlink)

        soup = BeautifulSoup(page.content, "html.parser")

        temp_list = []

        for tr_tag in soup.find_all("tr", attrs={"class": "fact_row"}):
            td_tags = tr_tag.find_all("td")

            for td_tag in td_tags:
                try:
                    temp_list.append(td_tag.find_all("div", attrs={"class": "value"})[0].contents[0])
                except:
                    temp_list.append("")

        new_planets_data.append(temp_list)

    except:
        time.sleep(1)
        scrape_more_data(hyperlink)
```

	Let's call the method and check the list new_planets_data.	

```
for index, data in enumerate(planets_data):
    scrape_more_data(data[5])
    print(f"scraping at {index+1} is completed.")

print(new_planets_data[0:10])
```

	Save the code and run using virtual environment.	

		
	Since we are running it for only four pages, it will start scraping the data from respective hyperlink.	
		
	Great job! Now we have 2 lists, **planets_data** and **new_planets_data**. In **new_planets_data**, a special character '\n' is present. we need to remove it before saving it to csv file. Also, we want to merge the two lists. Adding 2 lists creates 1 final list with elements from both the lists in the same order.	*The student merges the data.*

```
final_planet_data = []

for index, data in enumerate(planets_data):
    new_planet_data_element = new_planets_data[index]
    new_planet_data_element = [elem.replace("\n", "") for elem in new_planet_data_element]
    new_planet_data_element = new_planet_data_element[:7]
    final_planet_data.append(data + new_planet_data_element)
```

	Finally, we will create a csv with our **headers** and **final_planet_data**.	*The student creates a CSV.*

```
with open("final.csv", "w") as f:
        csvwriter = csv.writer(f)
        csvwriter.writerow(headers)
        csvwriter.writerows(final_planet_data)
```

	Let's run this code to see if it works fine and generates the **final.csv.**	*Student runs the code.*

final.csv

```
name,light_years_from_earth,planet_mass,stellar_magnitude,discovery_date,hyperlink,planet_type,planet_radius,orbital_radius,orbital

11 Comae Berenices b,304,19.4 Jupiters,4.72307,2007,https://exoplanets.nasa.gov/exoplanet-catalog/6988/11-comae-berenices-b/,Gas Gi

11 Ursae Minoris b,409,14.74 Jupiters,5.013,2009,https://exoplanets.nasa.gov/exoplanet-catalog/6989/11-ursae-minoris-b/,Gas Giant,2

14 Andromedae b,246,4.8 Jupiters,5.23133,2008,https://exoplanets.nasa.gov/exoplanet-catalog/6990/14-andromedae-b/,Gas Giant,2008,4.

14 Herculis b,58,4.66 Jupiters,6.61935,2002,https://exoplanets.nasa.gov/exoplanet-catalog/6991/14-herculis-b/,Gas Giant,2002,4.66 J

16 Cygni B b,69,1.78 Jupiters,6.215,1996,https://exoplanets.nasa.gov/exoplanet-catalog/6992/16-cygni-b-b/,Gas Giant,1996,1.78 Jupit

17 Scorpii b,408,4.32 Jupiters,5.22606,2020,https://exoplanets.nasa.gov/exoplanet-catalog/8016/17-scorpii-b/,Gas Giant,2020,4.32 Ju

18 Delphini b,249,10.3 Jupiters,5.51048,2008,https://exoplanets.nasa.gov/exoplanet-catalog/6993/18-delphini-b/,Gas Giant,2008,10.3

1RXS J160929.1-210524 b,454,8 Jupiters,12.618,2008,https://exoplanets.nasa.gov/exoplanet-catalog/7061/1rxs-j1609291-210524-b/,Gas G

24 Bootis b,313,0.91 Jupiters,5.59,2018,https://exoplanets.nasa.gov/exoplanet-catalog/7274/24-bootis-b/,Gas Giant,2018,0.91 Jupiter

24 Sextantis b,235,1.99 Jupiters,6.4535,2010,https://exoplanets.nasa.gov/exoplanet-catalog/6994/24-sextantis-b/,Gas Giant,2010,1.9

24 Sextantis c,235,0.86 Jupiters,6.4535,2010,https://exoplanets.nasa.gov/exoplanet-catalog/6995/24-sextantis-c/,Gas Giant,2010,0.8
```

	Although we have only scraped 4 pages, there are approximately 200 pages with 25 planets on each page. The scraping can take a lot of time sometimes (like for scraping 5000 planets data in this case) therefore we'll provide you the **final.csv** with the data of all the exoplanets. <Student Activity 1> https://github.com/procodingclass/PRO-129-Datasets If you want you can also try running your code after the class to check the output	Student runs the code after class to get the output or downloads the csv from Student Activity 1
Teacher Guides Student to Stop Screen Share		
FEEDBACK • **Appreciate the student for their efforts** • **Identify 2 strengths and 1 area of progress for the student**		
Step 4: Wrap-Up (5 min)	So, in this project class we revisited the concepts from the previous class and you did the majority of the scraping yourself! Congratulations!	**ESR:** Thanks!
	Next class, we will be learning new concepts and building new projects.	-
Teacher Clicks ✖ End Class		

Activity	Activity Name	Links
Teacher activity 1	solution	https://github.com/procodingclass/PRO-C128-RefCode
Student Activity 1	final csv	https://github.com/procodingclass/PRO-129-Datasets