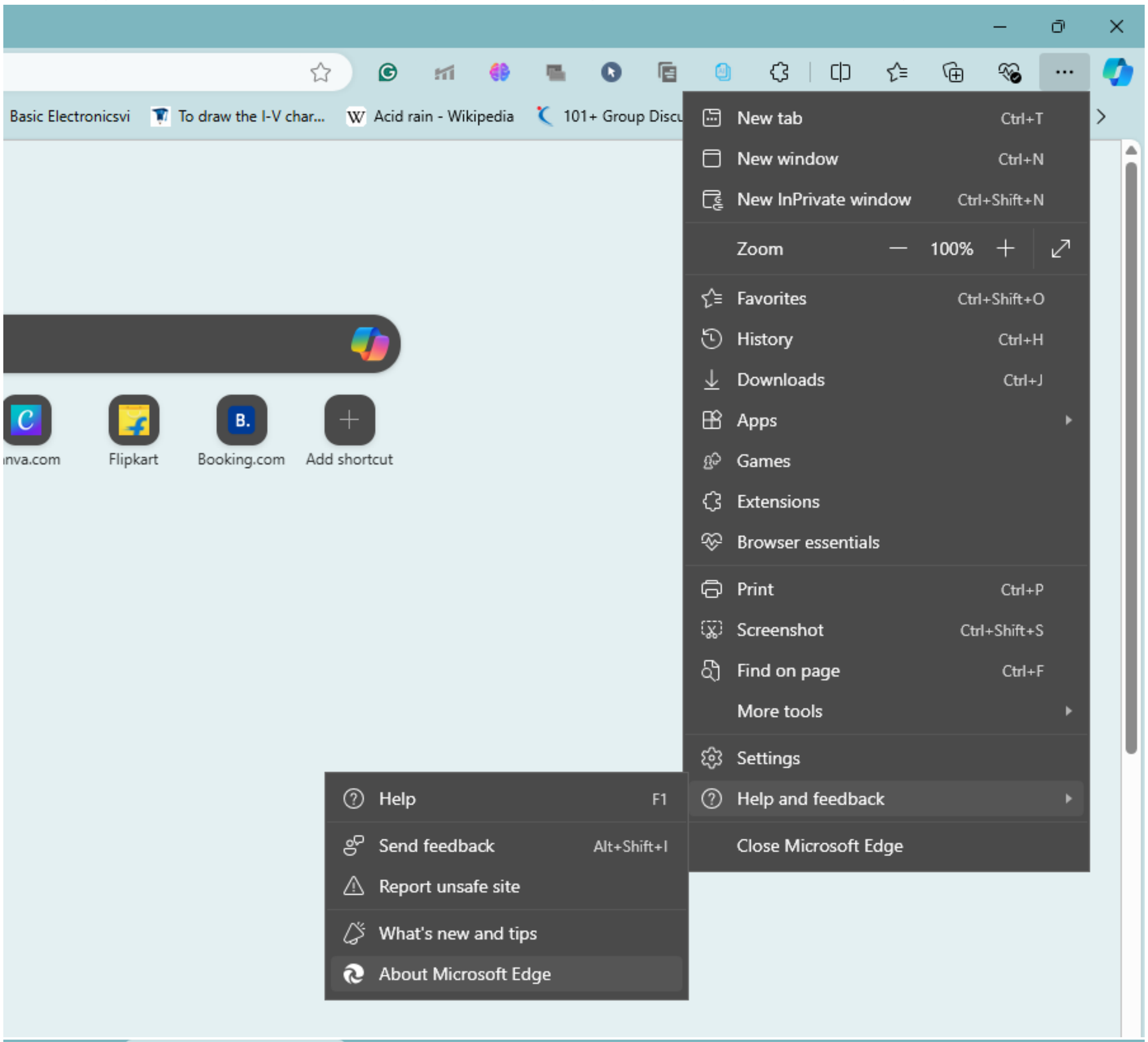
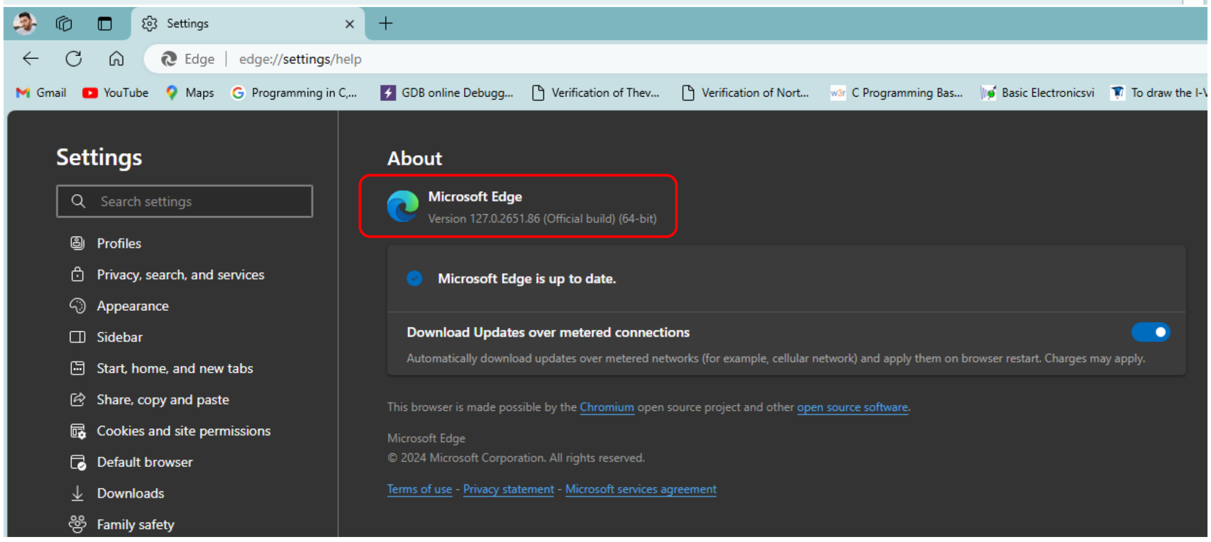
Installation of Selenium for Web Scrapping.

1. Check the edge browser version.
   Open edge → click on 3 dots on top right corner → go to help and feedback → go to about edge section





About

Microsoft Edge
Version 127.0.2651.86 (Official build) (64-bit)

Microsoft Edge is up to date.

Download Updates over metered connections
Automatically download updates over metered networks (for example, cellular network) and apply them on browser restart. Charges may apply.

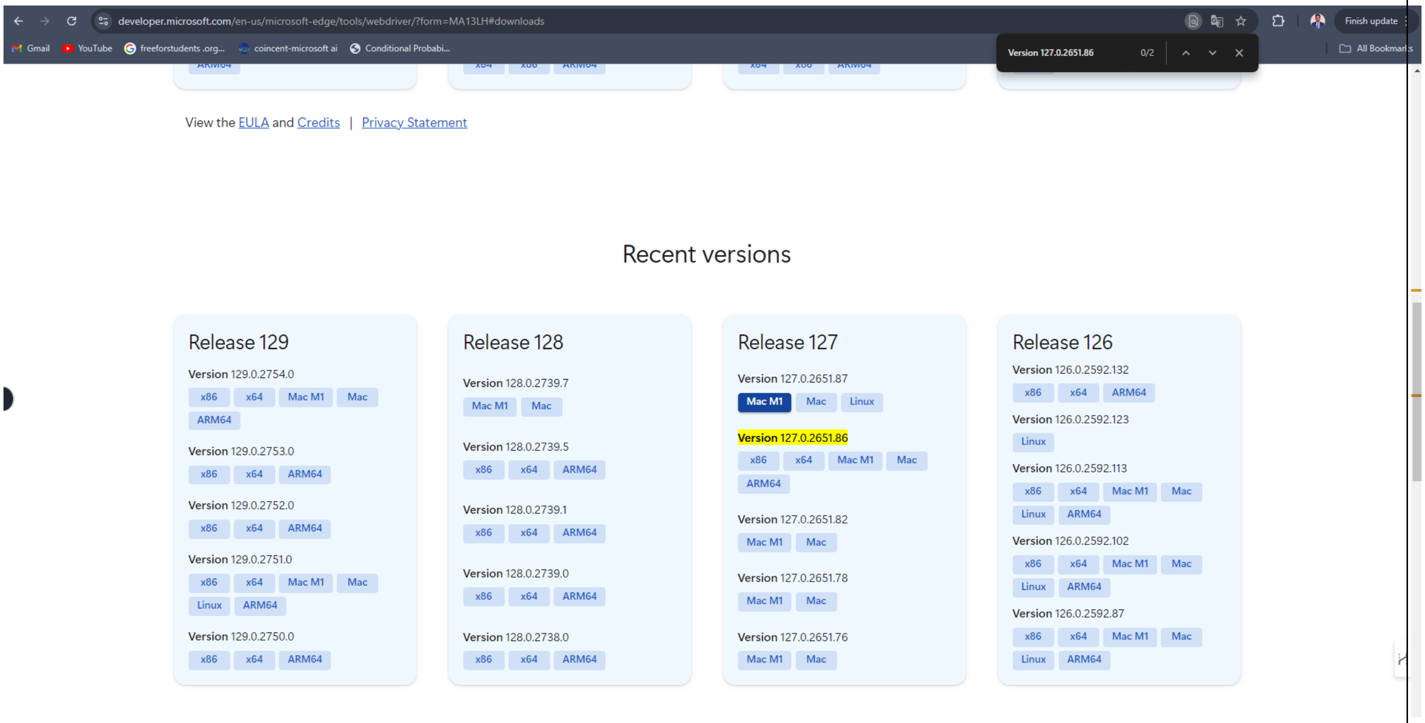
This browser is made possible by the Chromium open source project and other open source software.

Microsoft Edge
© 2024 Microsoft Corporation. All rights reserved.

Terms of use - Privacy statement - Microsoft services agreement

2. Copy the version …
3. Click on this link:
   https://developer.microsoft.com/en-us/microsoft-edge/tools/webdriver/?form=MA13LH#downloads
4. You will navigate to a selenium edgewebdriver page..

5. Ctrl + F and paste the version


developer.microsoft.com/en-us/microsoft-edge/tools/webdriver/?form=MA13LH#downloads

Gmail YouTube freeforstudents.org... coincent-microsoft ai Conditional Probabi... All Bookmarks

Finish update

Version 127.0.2651.86 0/2

View the EULA and Credits | Privacy Statement

Recent versions

Release 129

Version 129.0.2754.0
x86 x64 Mac M1 Mac ARM64

Version 129.0.2753.0
x86 x64 ARM64

Version 129.0.2752.0
x86 x64 ARM64

Version 129.0.2751.0
x86 x64 Mac M1 Mac Linux ARM64

Version 129.0.2750.0
x86 x64 ARM64

Release 128

Version 128.0.2739.7
Mac M1 Mac

Version 128.0.2739.5
x86 x64 ARM64

Version 128.0.2739.1
x86 x64 ARM64

Version 128.0.2739.0
x86 x64 ARM64

Version 128.0.2738.0
x86 x64 ARM64

Release 127

Version 127.0.2651.87
Mac M1 Mac Linux

Version 127.0.2651.86
x86 x64 Mac M1 Mac ARM64

Version 127.0.2651.82
Mac M1 Mac

Version 127.0.2651.78
Mac M1 Mac

Version 127.0.2651.76
Mac M1 Mac

Release 126

Version 126.0.2592.132
x86 x64 ARM64

Version 126.0.2592.123
Linux

Version 126.0.2592.113
x86 x64 Mac M1 Mac Linux ARM64

Version 126.0.2592.102
x86 x64 Mac M1 Mac Linux ARM64

Version 126.0.2592.87
x86 x64 Mac M1 Mac Linux ARM64

6. Download the same version of edge browser for your system requirements…
7. Once downloaded.. Extract the zip file and copy .exe file path

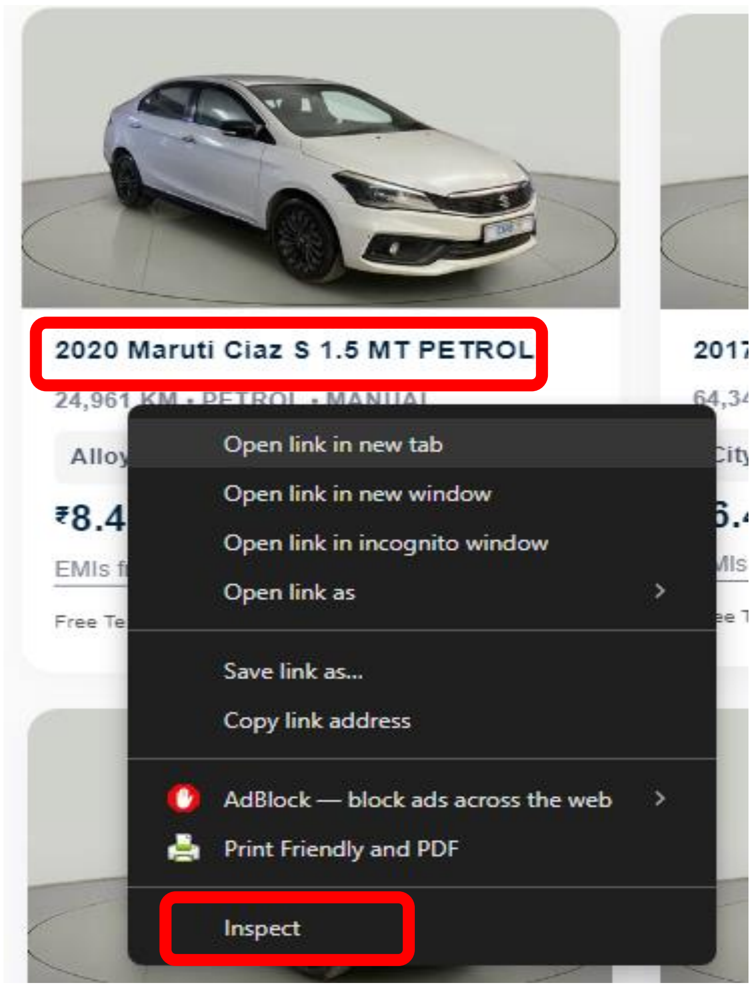
Selenium installation for python

1. pip install selenium
2. once installed…

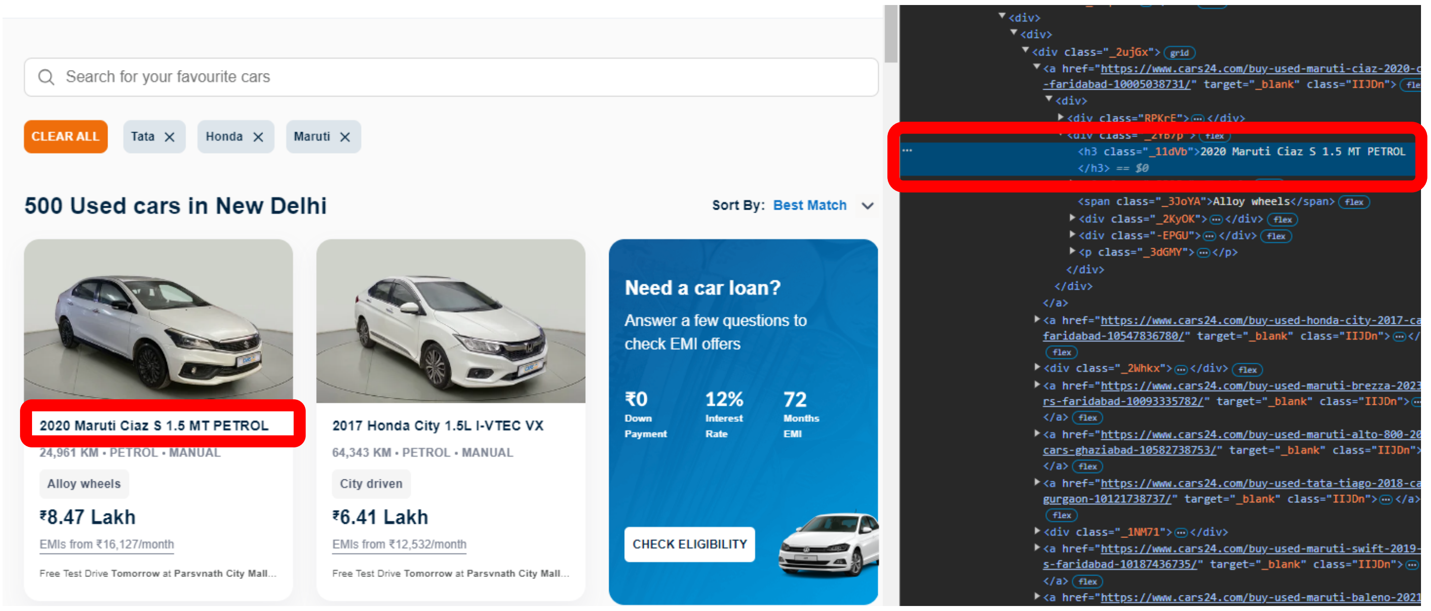
basic syntax:

```
from selenium import webdriver

from selenium.webdriver.edge.service import Service

link = “website llink”

# Specify the path to the Edge WebDriver executable

driver_path = "Paste the .exe file path here”


# Create a Service object with the executable path

service = Service(driver_path)


# Set up the Edge WebDriver using the Service object

driver = webdriver.Edge(service=service)


# Navigate to website

driver.get(link)
```

How to see the particular attribute code while scrapping the web?



1. Choose the attribute and right click on it … you can see “inspect”. Click on it. Source code will display on the left hand side .


Search for your favourite cars
CLEAR ALL
Tata
Honda
Maruti
500 Used cars in New Delhi
Sort By: Best Match
2020 Maruti Ciaz S 1.5 MT PETROL
24,961 KM • PETROL • MANUAL
Alloy wheels
₹8.47 Lakh
EMIs from ₹16,127/month
Free Test Drive Tomorrow at Parsvnath City Mall...
2017 Honda City 1.5L I-VTEC VX
64,343 KM • PETROL • MANUAL
City driven
₹6.41 Lakh
EMIs from ₹12,532/month
Free Test Drive Tomorrow at Parsvnath City Mall...
Need a car loan?
Answer a few questions to check EMI offers
₹0 Down Payment
12% Interest Rate
72 Months EMI
CHECK ELIGIBILITY

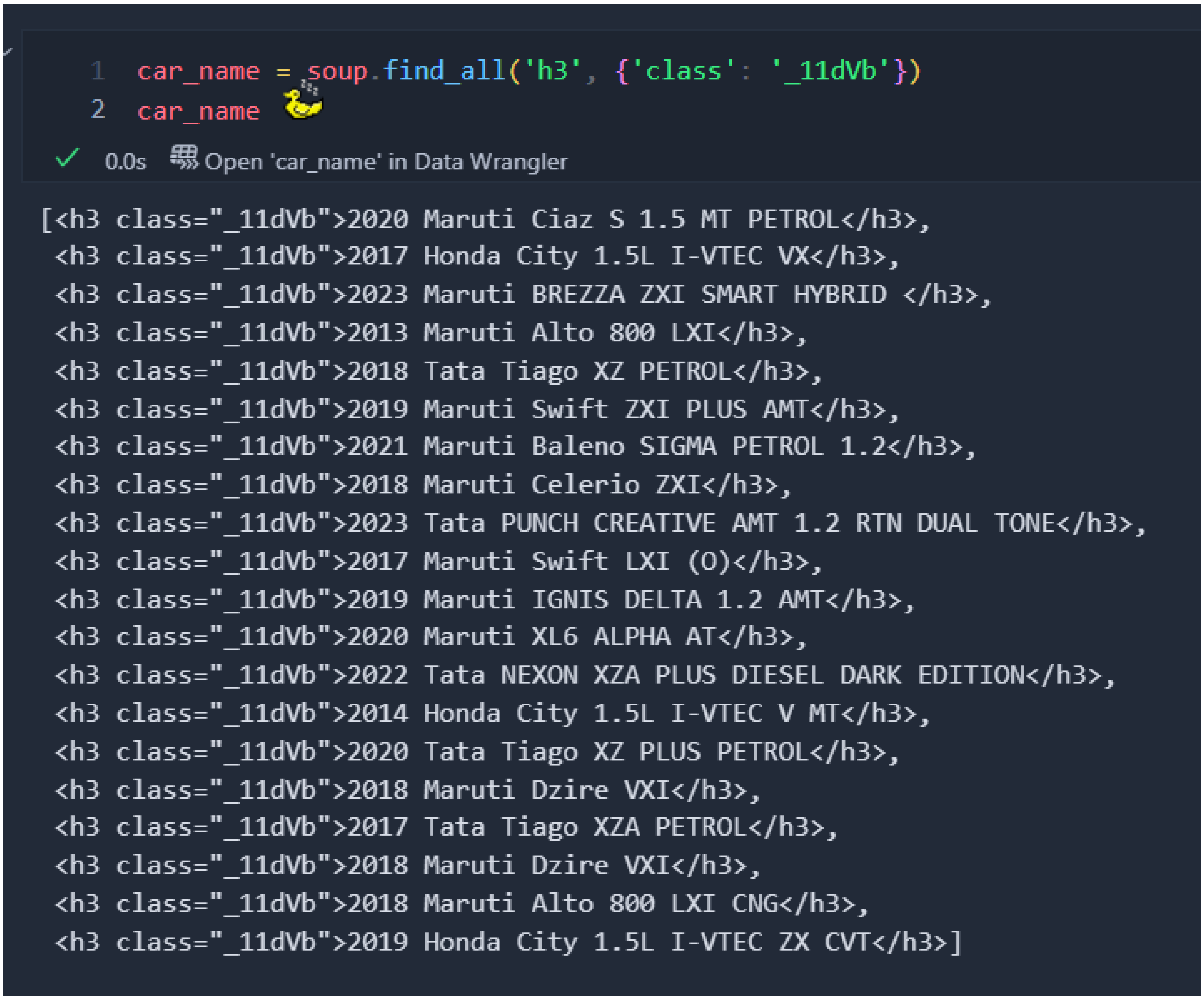
```
▼<div>
 ▼<div>
  ▼<div class="_2ujGx"> grid
   ▼<a href="https://www.cars24.com/buy-used-maruti-ciaz-2020-c
    -faridabad-10005038731/" target="_blank" class="IIJDn"> fle
    ▼<div>
     ▶<div class="RPKrE"> ⋯ </div>
      <div class="_2YB7p"> flex
        <h3 class="_11dVb">2020 Maruti Ciaz S 1.5 MT PETROL
        </h3> == $0
        <span class="_3JoYA">Alloy wheels</span> flex
       ▶<div class="_2KyOK"> ⋯ </div> flex
       ▶<div class="-EPGU"> ⋯ </div> flex
       ▶<p class="_3dGMY"> ⋯ </p>
      </div>
     </div>
    </a>
   ▶<a href="https://www.cars24.com/buy-used-honda-city-2017-ca
    faridabad-10547836780/" target="_blank" class="IIJDn"> ⋯ </
    flex
   ▶<div class="_2Whkx"> ⋯ </div> flex
   ▶<a href="https://www.cars24.com/buy-used-maruti-brezza-2023
    rs-faridabad-10093335782/" target="_blank" class="IIJDn"> ⋯
    </a> flex
   ▶<a href="https://www.cars24.com/buy-used-maruti-alto-800-20
    cars-ghaziabad-10582738753/" target="_blank" class="IIJDn">
    </a> flex
   ▶<a href="https://www.cars24.com/buy-used-tata-tiago-2018-ca
    gurgaon-10121738737/" target="_blank" class="IIJDn"> ⋯ </a>
    flex
   ▶<div class="_1NM71"> ⋯ </div>
   ▶<a href="https://www.cars24.com/buy-used-maruti-swift-2019-
    s-faridabad-10187436735/" target="_blank" class="IIJDn"> ⋯
    </a> flex
   ▶<a href="https://www.cars24.com/buy-used-maruti-baleno-2021
```

We are trying to scrape name of the car. I inspect the name as I mentioned above. You can see that a piece of code is highlighted in blue colour.

2. Copy the class name from the highlighted code
3. And check the tag name like h3/div/img/span/p/a/url
   We need these names to specify which tag with which class name our code should scrape.
4. Basic code for web scrapping using BeautifulSoup:

```
from bs4 import BeautifulSoup
import requests

# URL of the page to scrape
url = 'https://example.com'

# Send a GET request to the URL
response = requests.get(url)

# Parse the page content
soup = BeautifulSoup(response.content, 'html.parser')

# Define the tag and class name to scrape
tag = 'div'
class_name = 'example-class'

# Find all elements with the specified tag and class name
elements = soup.find_all(tag, class_=class_name)

# Print the text content of each element
for element in elements:
    print(element.get_text())
```

5. Code for scrapping the name of the car



```python
car_name = soup.find_all('h3', {'class': '_11dVb'})
car_name
```

```
[<h3 class="_11dVb">2020 Maruti Ciaz S 1.5 MT PETROL</h3>,
 <h3 class="_11dVb">2017 Honda City 1.5L I-VTEC VX</h3>,
 <h3 class="_11dVb">2023 Maruti BREZZA ZXI SMART HYBRID </h3>,
 <h3 class="_11dVb">2013 Maruti Alto 800 LXI</h3>,
 <h3 class="_11dVb">2018 Tata Tiago XZ PETROL</h3>,
 <h3 class="_11dVb">2019 Maruti Swift ZXI PLUS AMT</h3>,
 <h3 class="_11dVb">2021 Maruti Baleno SIGMA PETROL 1.2</h3>,
 <h3 class="_11dVb">2018 Maruti Celerio ZXI</h3>,
 <h3 class="_11dVb">2023 Tata PUNCH CREATIVE AMT 1.2 RTN DUAL TONE</h3>,
 <h3 class="_11dVb">2017 Maruti Swift LXI (O)</h3>,
 <h3 class="_11dVb">2019 Maruti IGNIS DELTA 1.2 AMT</h3>,
 <h3 class="_11dVb">2020 Maruti XL6 ALPHA AT</h3>,
 <h3 class="_11dVb">2022 Tata NEXON XZA PLUS DIESEL DARK EDITION</h3>,
 <h3 class="_11dVb">2014 Honda City 1.5L I-VTEC V MT</h3>,
 <h3 class="_11dVb">2020 Tata Tiago XZ PLUS PETROL</h3>,
 <h3 class="_11dVb">2018 Maruti Dzire VXI</h3>,
 <h3 class="_11dVb">2017 Tata Tiago XZA PETROL</h3>,
 <h3 class="_11dVb">2018 Maruti Dzire VXI</h3>,
 <h3 class="_11dVb">2018 Maruti Alto 800 LXI CNG</h3>,
 <h3 class="_11dVb">2019 Honda City 1.5L I-VTEC ZX CVT</h3>]
```

Explanation:

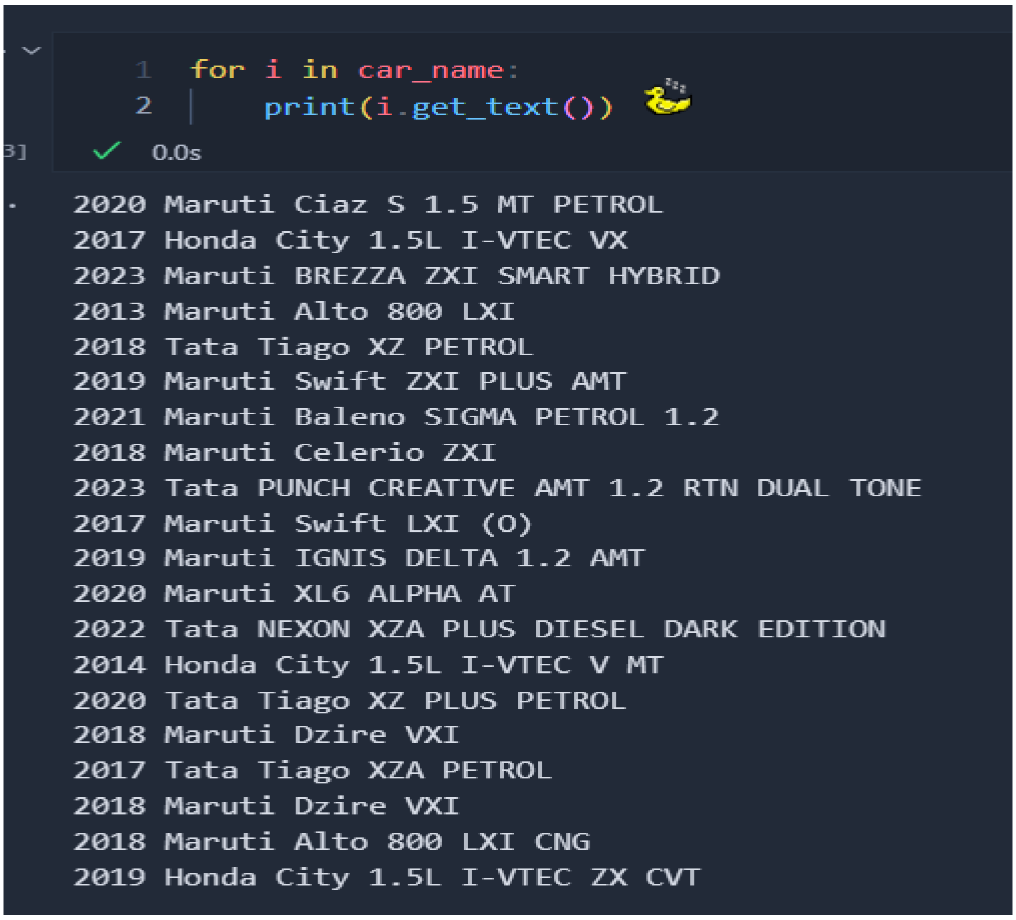
If you can observe the output it is saving in the form of list.

It’s because, we have a multiple values with h3 tag and _11dVb class.

We need only the textual content… not the html code.

We need to extract content from each value of the list.

So we iterate the whole list using for loop and get the text using get_text() function.

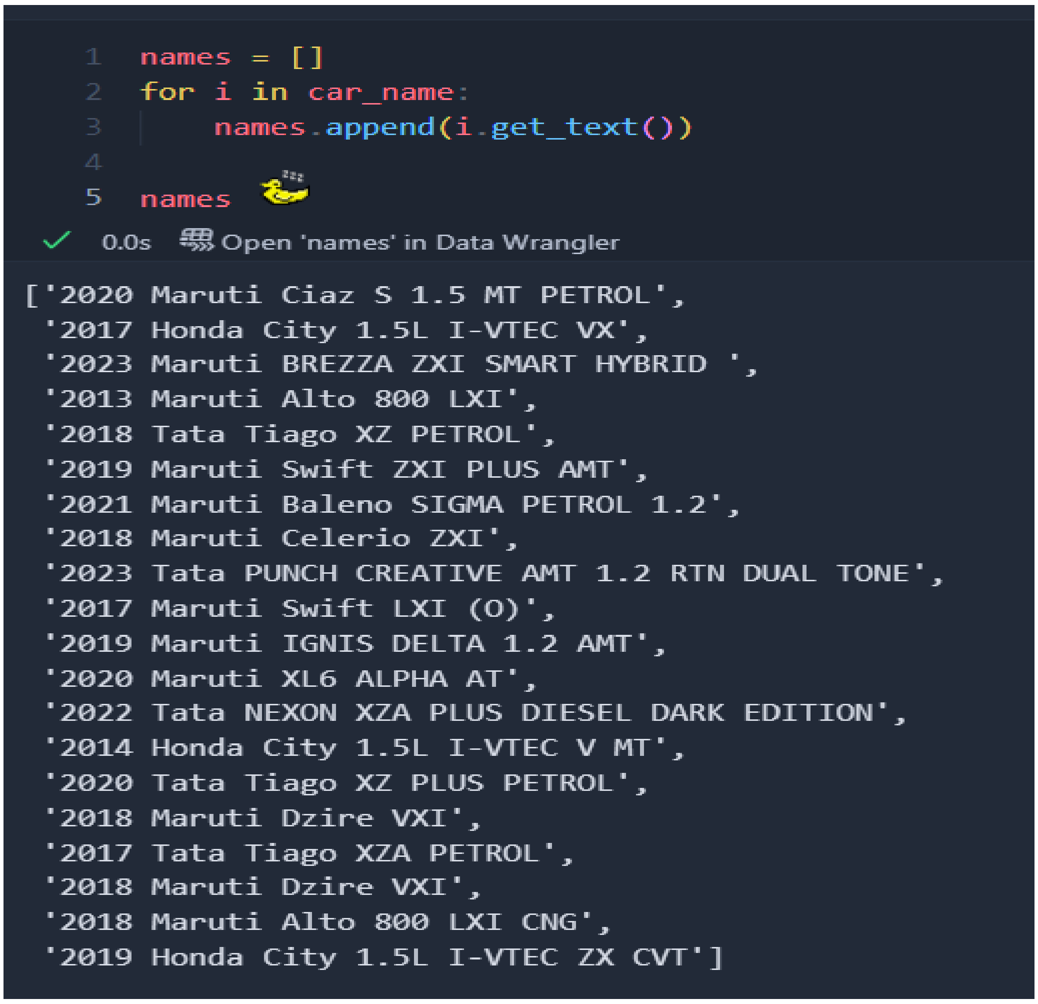


```
for i in car_name:
    print(i.get_text())
```

✓ 0.0s

```
2020 Maruti Ciaz S 1.5 MT PETROL
2017 Honda City 1.5L I-VTEC VX
2023 Maruti BREZZA ZXI SMART HYBRID
2013 Maruti Alto 800 LXI
2018 Tata Tiago XZ PETROL
2019 Maruti Swift ZXI PLUS AMT
2021 Maruti Baleno SIGMA PETROL 1.2
2018 Maruti Celerio ZXI
2023 Tata PUNCH CREATIVE AMT 1.2 RTN DUAL TONE
2017 Maruti Swift LXI (O)
2019 Maruti IGNIS DELTA 1.2 AMT
2020 Maruti XL6 ALPHA AT
2022 Tata NEXON XZA PLUS DIESEL DARK EDITION
2014 Honda City 1.5L I-VTEC V MT
2020 Tata Tiago XZ PLUS PETROL
2018 Maruti Dzire VXI
2017 Tata Tiago XZA PETROL
2018 Maruti Dzire VXI
2018 Maruti Alto 800 LXI CNG
2019 Honda City 1.5L I-VTEC ZX CVT
```

While we are working on any project. We need to store these values in the form of list. So first create a list and append the value iteratively.



```
names = []
for i in car_name:
    names.append(i.get_text())

names
```

✓ 0.0s Open 'names' in Data Wrangler

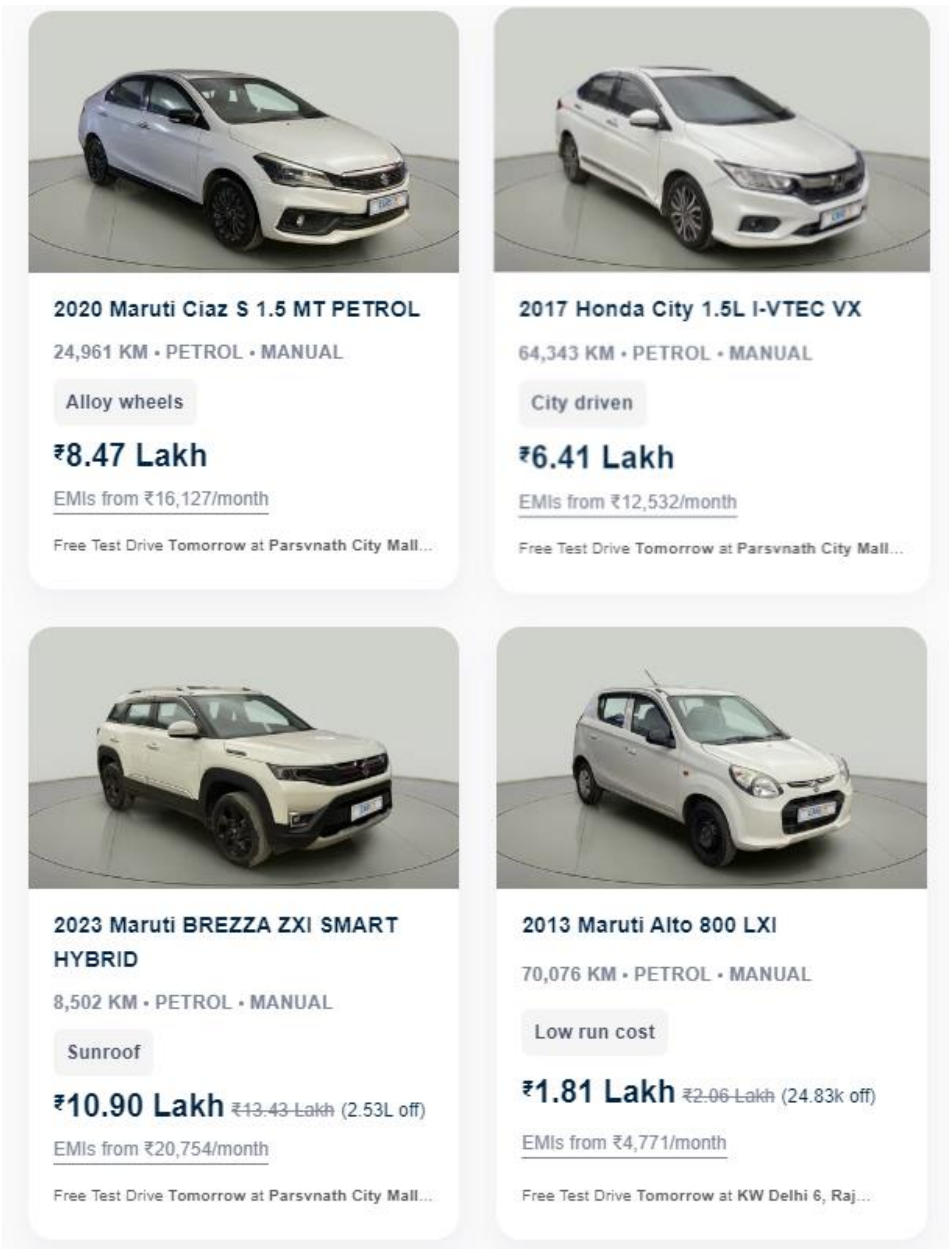
```
['2020 Maruti Ciaz S 1.5 MT PETROL',
 '2017 Honda City 1.5L I-VTEC VX',
 '2023 Maruti BREZZA ZXI SMART HYBRID ',
 '2013 Maruti Alto 800 LXI',
 '2018 Tata Tiago XZ PETROL',
 '2019 Maruti Swift ZXI PLUS AMT',
 '2021 Maruti Baleno SIGMA PETROL 1.2',
 '2018 Maruti Celerio ZXI',
 '2023 Tata PUNCH CREATIVE AMT 1.2 RTN DUAL TONE',
 '2017 Maruti Swift LXI (O)',
 '2019 Maruti IGNIS DELTA 1.2 AMT',
 '2020 Maruti XL6 ALPHA AT',
 '2022 Tata NEXON XZA PLUS DIESEL DARK EDITION',
 '2014 Honda City 1.5L I-VTEC V MT',
 '2020 Tata Tiago XZ PLUS PETROL',
 '2018 Maruti Dzire VXI',
 '2017 Tata Tiago XZA PETROL',
 '2018 Maruti Dzire VXI',
 '2018 Maruti Alto 800 LXI CNG',
 '2019 Honda City 1.5L I-VTEC ZX CVT']
```

Sometimes we have a situation like… some cars in the website doesn't have names for it.

That's what we call as missing values…Let's learn how to handle it

Here we use **if else conditions**…

- **If there is content in the tag do extract text operation or else 'N/A' .. it stores Null value if the values are not available.**



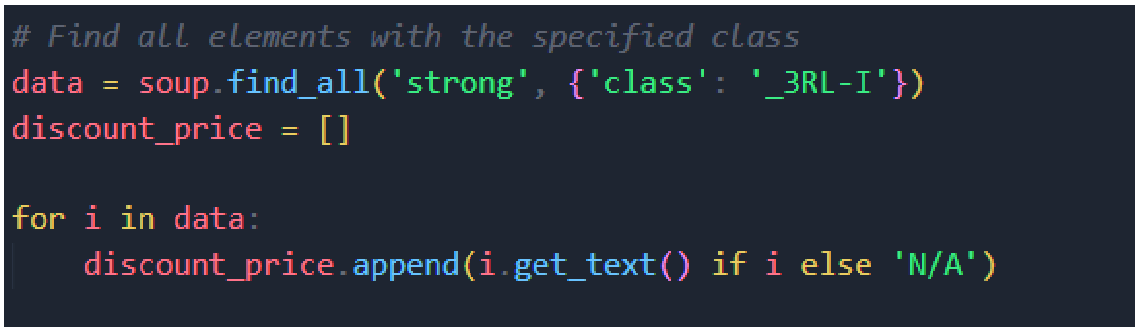
2020 Maruti Ciaz S 1.5 MT PETROL
24,961 KM • PETROL • MANUAL
Alloy wheels
₹8.47 Lakh
EMIs from ₹16,127/month
Free Test Drive Tomorrow at Parsvnath City Mall...

2017 Honda City 1.5L I-VTEC VX
64,343 KM • PETROL • MANUAL
City driven
₹6.41 Lakh
EMIs from ₹12,532/month
Free Test Drive Tomorrow at Parsvnath City Mall...

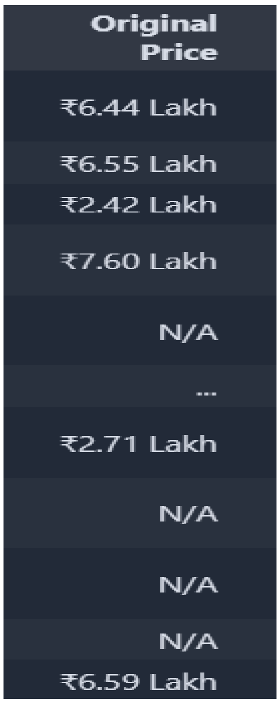
2023 Maruti BREZZA ZXI SMART HYBRID
8,502 KM • PETROL • MANUAL
Sunroof
₹10.90 Lakh ~~₹13.43 Lakh~~ (2.53L off)
EMIs from ₹20,754/month
Free Test Drive Tomorrow at Parsvnath City Mall...

2013 Maruti Alto 800 LXI
70,076 KM • PETROL • MANUAL
Low run cost
₹1.81 Lakh ~~₹2.06 Lakh~~ (24.83k off)
EMIs from ₹4,771/month
Free Test Drive Tomorrow at KW Delhi 6, Raj...

If you see this picture… two cars having discount price and two cars doesn't have discount price. Let's handle this situation with if else condition.



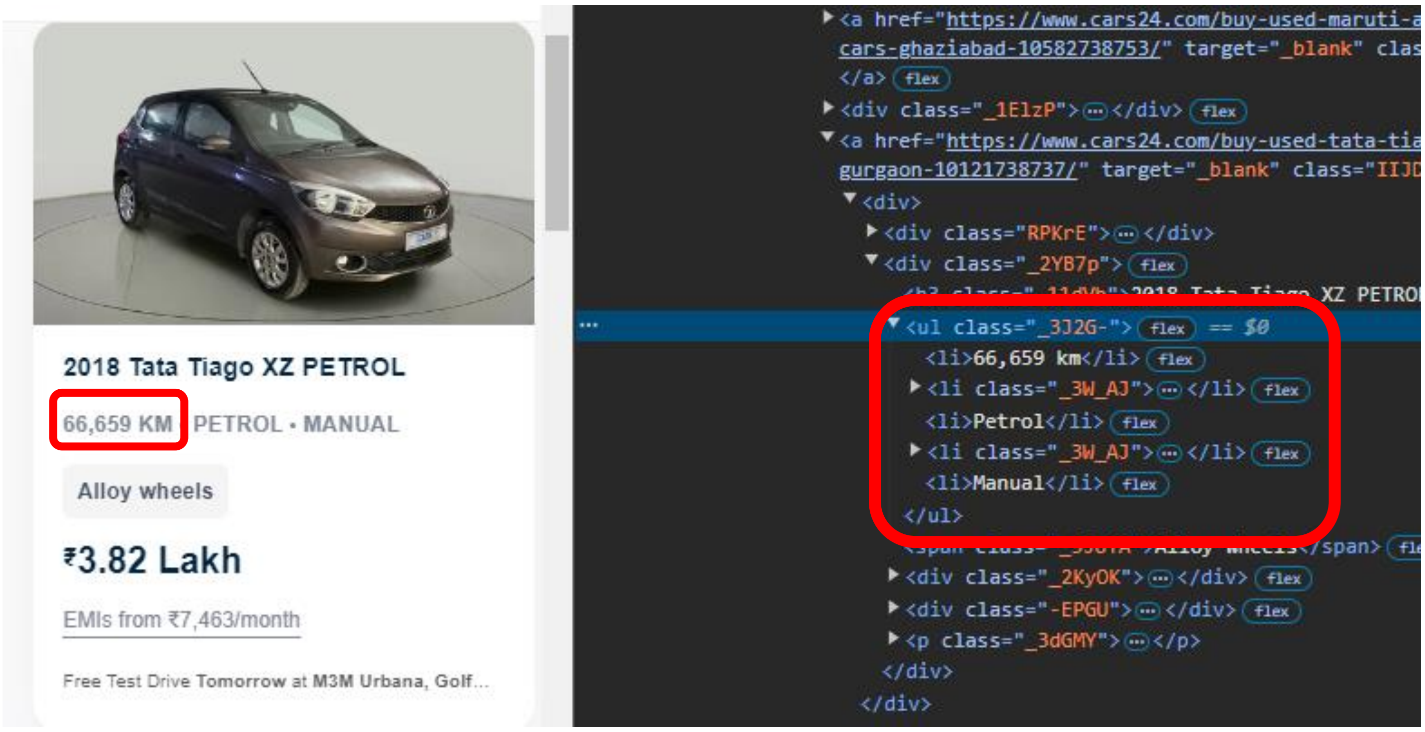
```python
# Find all elements with the specified class
data = soup.find_all('strong', {'class': '_3RL-I'})
discount_price = []

for i in data:
    discount_price.append(i.get_text() if i else 'N/A')
```



Original Price
₹6.44 Lakh
₹6.55 Lakh
₹2.42 Lakh
₹7.60 Lakh
N/A
...
₹2.71 Lakh
N/A
N/A
N/A
₹6.59 Lakh

You can see the output right side.. some column are having N/A .. indicates they have null values..

Sometimes, The values we want to scrape is in the form of lists(html lsits with li tags).



2018 Tata Tiago XZ PETROL

66,659 KM PETROL • MANUAL

Alloy wheels

₹3.82 Lakh

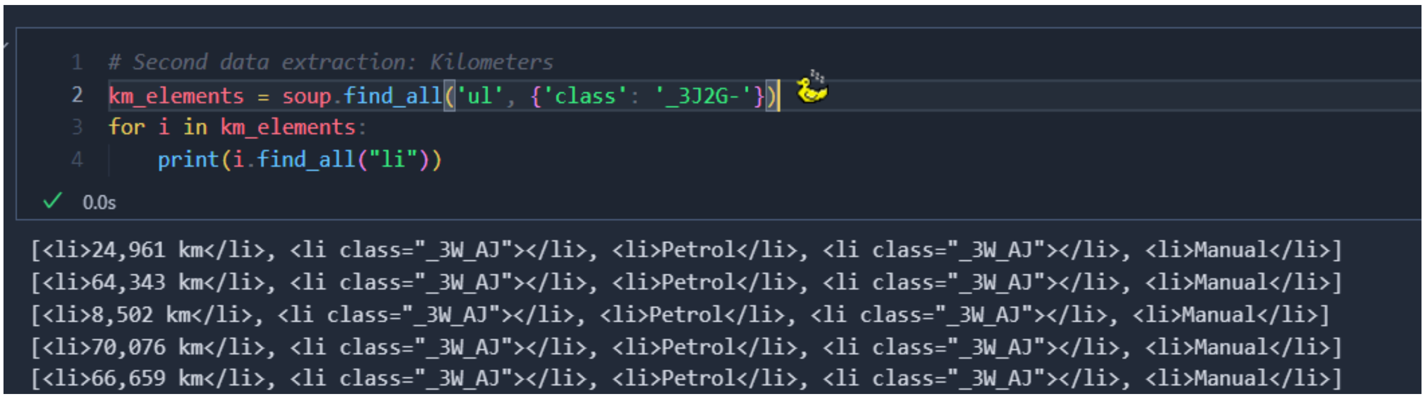
EMIs from ₹7,463/month

Free Test Drive Tomorrow at M3M Urbana, Golf...

```html
▶<a href="https://www.cars24.com/buy-used-maruti-a
cars-ghaziabad-10582738753/" target="_blank" clas
</a> flex
▶<div class="_1ElzP">⋯</div> flex
▼<a href="https://www.cars24.com/buy-used-tata-tia
gurgaon-10121738737/" target="_blank" class="IIJD
  ▼<div>
    ▶<div class="RPKrE">⋯</div>
    ▼<div class="_2YB7p"> flex
        <h3 class="_11dVb">2018 Tata Tiago XZ PETROL
...    ▼<ul class="_3J2G-"> flex == $0
          <li>66,659 km</li> flex
         ▶<li class="_3W_AJ">⋯</li> flex
          <li>Petrol</li> flex
         ▶<li class="_3W_AJ">⋯</li> flex
          <li>Manual</li> flex
        </ul>
        <span class="_3JGYA">Alloy wheels</span> fl
       ▶<div class="_2KyOK">⋯</div> flex
       ▶<div class="-EPGU">⋯</div> flex
       ▶<p class="_3dGMY">⋯</p>
      </div>
    </div>
```

I want to scrape km from the website.

1. Firstly, we don't have a specific class to scrape km.
2. We only have class for **ul tag.**
3. Copy the ul tag class name..



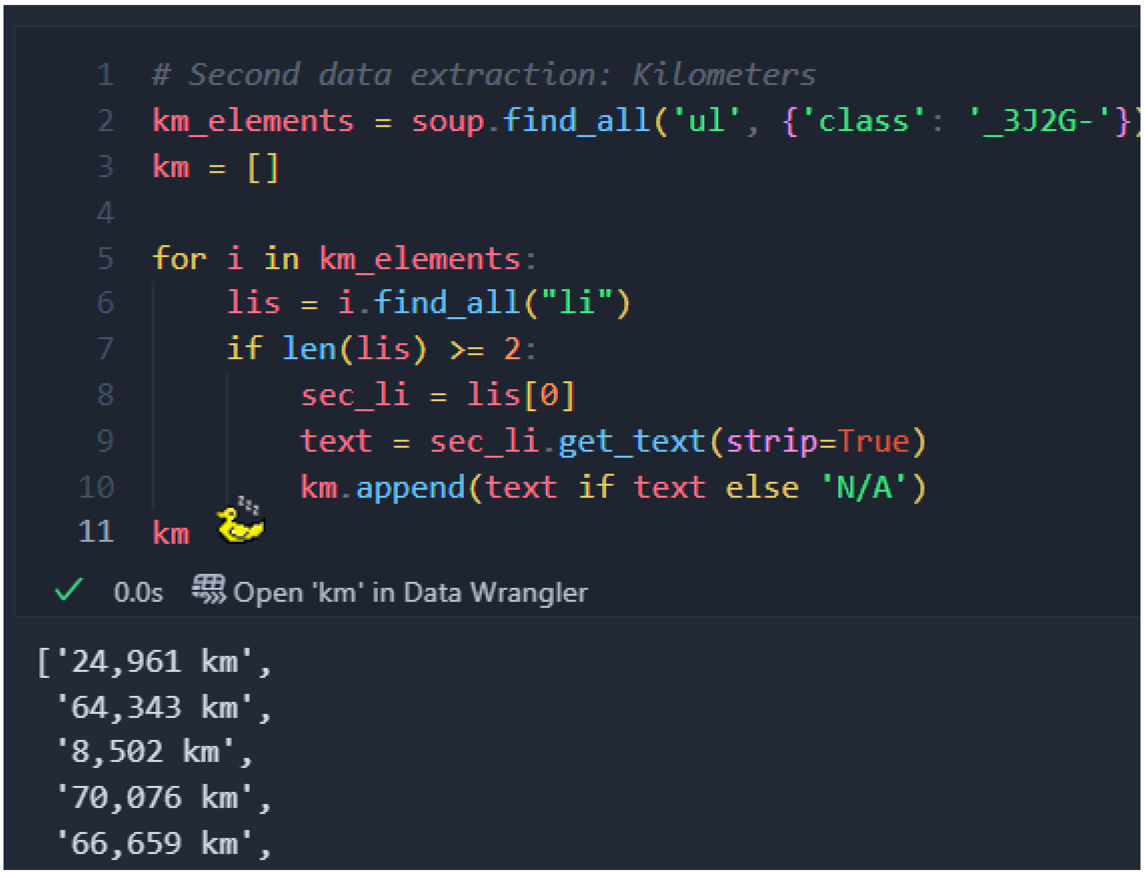
```python
# Second data extraction: Kilometers
km_elements = soup.find_all('ul', {'class': '_3J2G-'})
for i in km_elements:
    print(i.find_all("li"))
```

```
✓ 0.0s
[<li>24,961 km</li>, <li class="_3W_AJ"></li>, <li>Petrol</li>, <li class="_3W_AJ"></li>, <li>Manual</li>]
[<li>64,343 km</li>, <li class="_3W_AJ"></li>, <li>Petrol</li>, <li class="_3W_AJ"></li>, <li>Manual</li>]
[<li>8,502 km</li>, <li class="_3W_AJ"></li>, <li>Petrol</li>, <li class="_3W_AJ"></li>, <li>Manual</li>]
[<li>70,076 km</li>, <li class="_3W_AJ"></li>, <li>Petrol</li>, <li class="_3W_AJ"></li>, <li>Manual</li>]
[<li>66,659 km</li>, <li class="_3W_AJ"></li>, <li>Petrol</li>, <li class="_3W_AJ"></li>, <li>Manual</li>]
```

You can see the output with nested list…

And every value is under li tag itself… In this situation we select our attribute using indexing.

We can see that km in 0th index.



```python
# Second data extraction: Kilometers
km_elements = soup.find_all('ul', {'class': '_3J2G-'})
km = []

for i in km_elements:
    lis = i.find_all("li")
    if len(lis) >= 2:
        sec_li = lis[0]
        text = sec_li.get_text(strip=True)
        km.append(text if text else 'N/A')
km
```

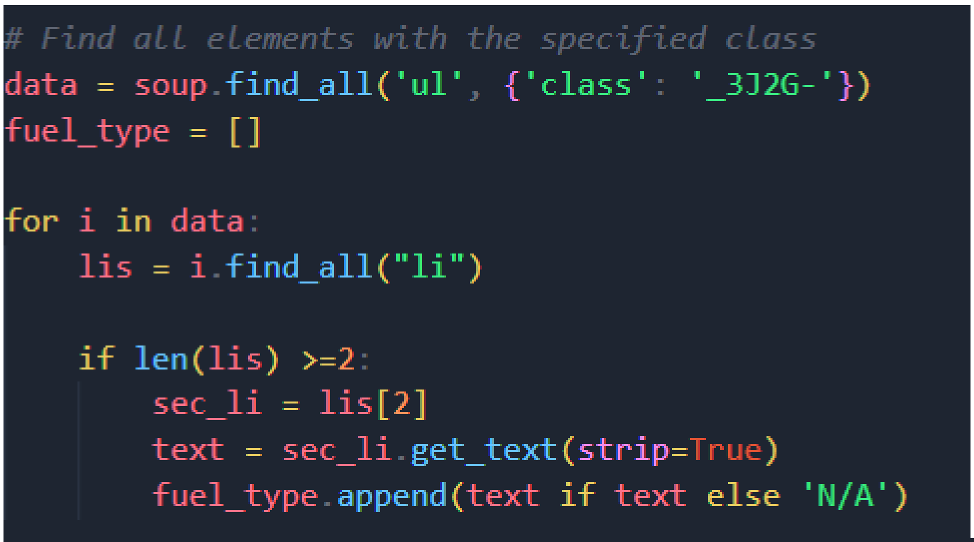
```
['24,961 km',
 '64,343 km',
 '8,502 km',
 '70,076 km',
 '66,659 km',
```

So, we specified as lis[0] to get the values of 0th index of every inner list from nested lists.

And we use if else .. for null values handling

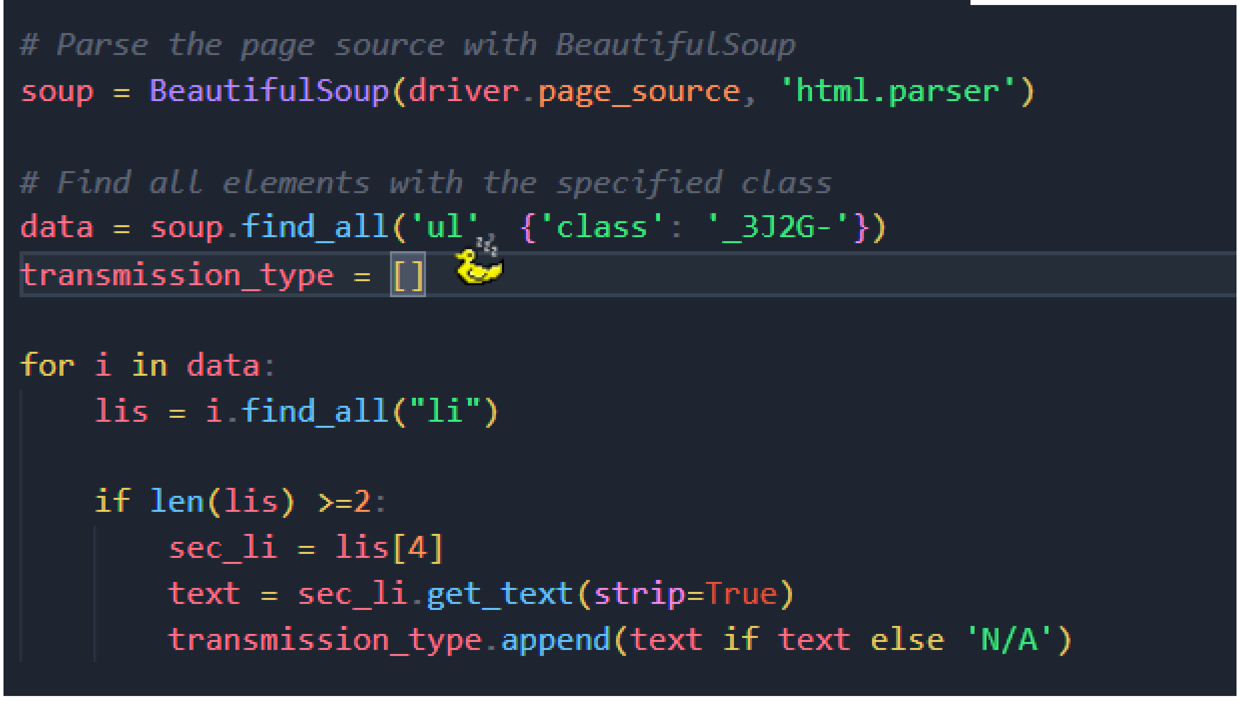
If we observe the nested list, at index 1 and index 3 we have dummy values..

So for accessing fuel_type and transmission_type we need skip one index



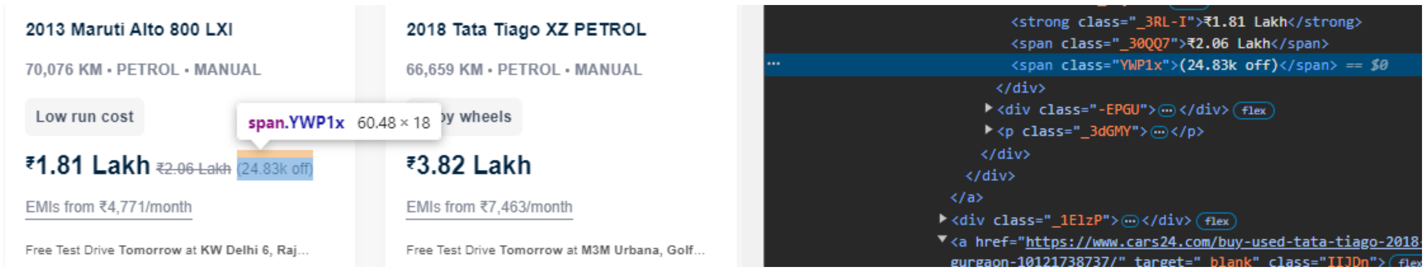
```python
# Find all elements with the specified class
data = soup.find_all('ul', {'class': '_3J2G-'})
fuel_type = []

for i in data:
    lis = i.find_all("li")

    if len(lis) >=2:
        sec_li = lis[2]
        text = sec_li.get_text(strip=True)
        fuel_type.append(text if text else 'N/A')
```

we use 2 because we need to skip 1st index



```python
# Parse the page source with BeautifulSoup
soup = BeautifulSoup(driver.page_source, 'html.parser')

# Find all elements with the specified class
data = soup.find_all('ul', {'class': '_3J2G-'})
transmission_type = []

for i in data:
    lis = i.find_all("li")

    if len(lis) >=2:
        sec_li = lis[4]
        text = sec_li.get_text(strip=True)
        transmission_type.append(text if text else 'N/A')
```

1we used index 4 , because we need to skip 3rd index

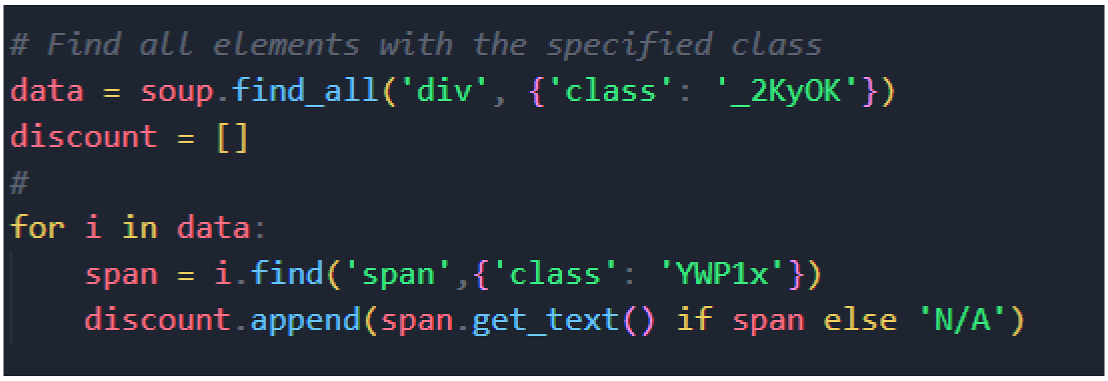


2013 Maruti Alto 800 LXI
70,076 KM • PETROL • MANUAL
Low run cost
₹1.81 Lakh ~~₹2.06 Lakh~~ (24.83k off)
EMIs from ₹4,771/month
Free Test Drive Tomorrow at KW Delhi 6, Raj...

2018 Tata Tiago XZ PETROL
66,659 KM • PETROL • MANUAL
₹3.82 Lakh
EMIs from ₹7,463/month
Free Test Drive Tomorrow at M3M Urbana, Golf...

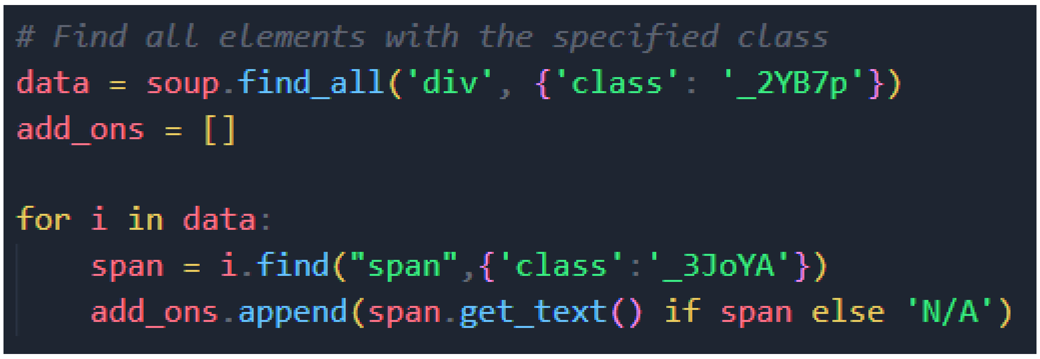
```
        <strong class="_3RL-I">₹1.81 Lakh</strong>
        <span class="_30QQ7">₹2.06 Lakh</span>
        <span class="YWP1x">(24.83k off)</span> == $0
      </div>
     ▸<div class="-EPGU">…</div> flex
     ▸<p class="_3dGMY">…</p>
    </div>
   </div>
  </a>
 ▸<div class="_1ElzP">…</div> flex
 ▾<a href="https://www.cars24.com/buy-used-tata-tiago-2018-
  gurgaon-10121738737/" target="_blank" class="IIJDn"> flex
```

Let’s see for offered price….it is in span tag and “YWP1x” class



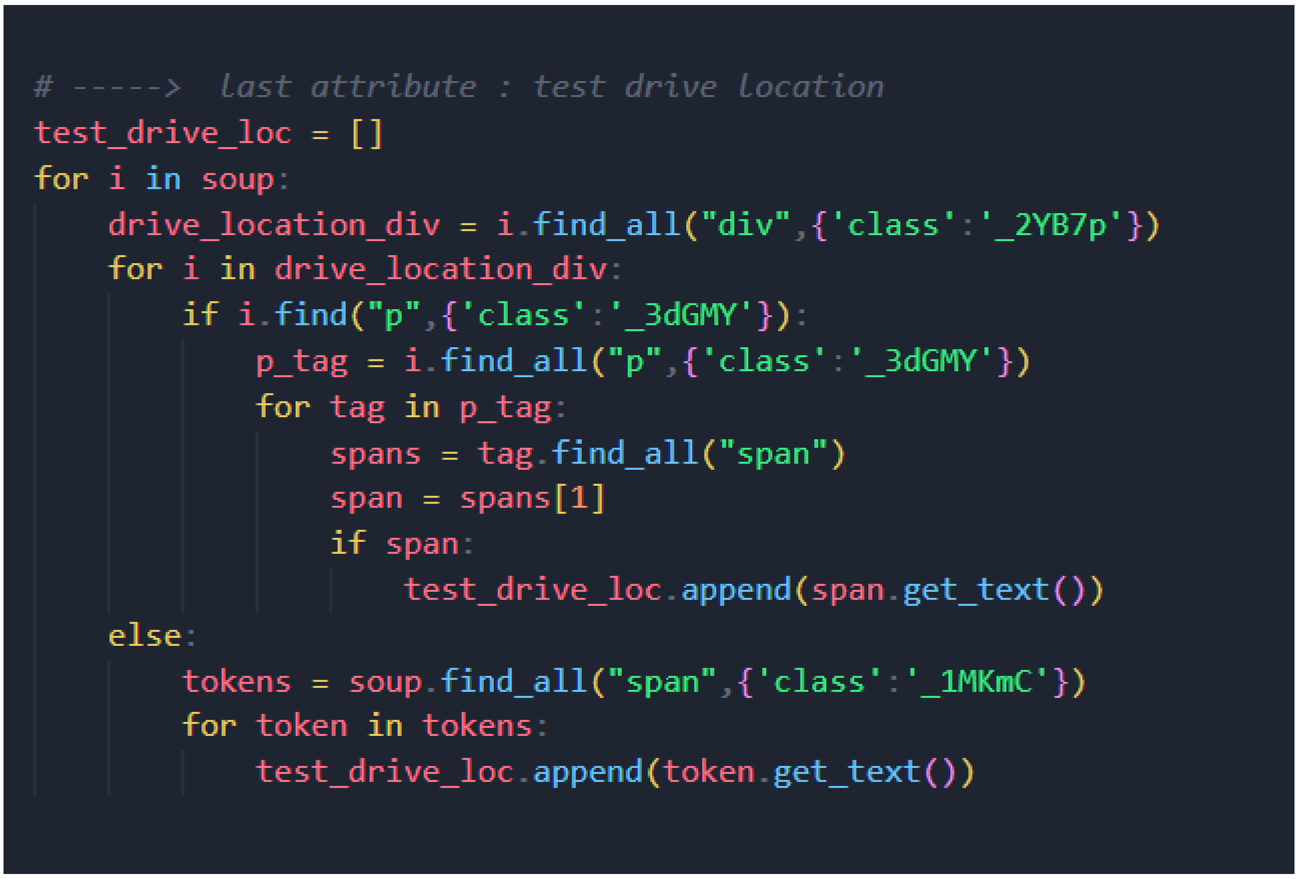
```python
# Find all elements with the specified class
data = soup.find_all('div', {'class': '_2KyOK'})
discount = []
#
for i in data:
    span = i.find('span',{'class': 'YWP1x'})
    discount.append(span.get_text() if span else 'N/A')
```

Same for add_ons…



```python
# Find all elements with the specified class
data = soup.find_all('div', {'class': '_2YB7p'})
add_ons = []

for i in data:
    span = i.find("span",{'class':'_3JoYA'})
    add_ons.append(span.get_text() if span else 'N/A')
```

Home work….



```python
# ----->  last attribute : test drive location
test_drive_loc = []
for i in soup:
    drive_location_div = i.find_all("div",{'class':'_2YB7p'})
    for i in drive_location_div:
        if i.find("p",{'class':'_3dGMY'}):
            p_tag = i.find_all("p",{'class':'_3dGMY'})
            for tag in p_tag:
                spans = tag.find_all("span")
                span = spans[1]
                if span:
                    test_drive_loc.append(span.get_text())
    else:
        tokens = soup.find_all("span",{'class':'_1MKmC'})
        for token in tokens:
            test_drive_loc.append(token.get_text())
```

I want you guys to go through with this code and explain in the grp… what you understood from it..
Try every code and convert the lists into dataframe and send the files in the grp…