jupyter Web_Scrapping Last Checkpoint: 4 minutes ago (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Scrapping truecar.com search_result by Kundan Pawar

Step 1 - Imports

In [1]:
```python
import requests
import pandas as pd
import sqlalchemy
import os

```

1. Go to truecar.com
2. Search according to preferences
3. When able to see the listings of elements page wise ,

> Step1
>
> > inspect the page element
> >
> > > Go to Network->Fetch/XHR
> > >
> > > > Navigate through the pages and clear previous response by "Stop" Symbol

> find the belonging Name of the resquest by navigating through their Json drop down menu
>
> > RTCLK the belonging Name and Copy>Copy_cURL(BASH)
> >
> > > Go to https://curlconverter.com/ and convert the cURL to cURL python and copy the code

And Paste the code below

In [2]:
```python

headers = {
    'authority': 'www.truecar.com',
    'accept': 'application/json, text/plain, */*',
    'accept-language': 'en-IN,en;q=0.7',
    # Requests sorts cookies= alphabetically
    # 'cookie': f"flag-abt-tcdc-about-us-page=true; flag-abt-hyundai-ioniq-5-experiment=true; flag-abt-true-car-plus-glol
    'if-none-match': 'W/"52db36c6f7181086af53306513cac330"',
    'referer': 'https://www.truecar.com/used-cars-for-sale/listings/price-10000-20000/location-toronto-ks/?page=2',
    'sec-fetch-dest': 'empty',
    'sec-fetch-mode': 'cors',
    'sec-fetch-site': 'same-origin',
    'sec-gpc': '1',
    'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/105.0.0.0 Safa
}

response = requests.get('https://www.truecar.com/abp/api/vehicles/used/listings?city=toronto&collapse=true&fallback=true&
# response = requests.get('https://www.truecar.com/abp/api/vehicles/used/listings?city=toronto&collapse=true&fallback=tru
```

check the status of response we get

In [3]:
```python
response

```

Out[3]:
```
<Response [200]>
```

Add variable to Json function of the response

In [4]:
```python
result_json = response.json()
```

find the drop down keys fo the variable_Of_Json_function

In [5]:
```python
result_json.keys()
```

Out[5]:
```
dict_keys(['listings', 'page', 'per_page', 'total', 'fallback', 'fallback_step', 'display', 'list_price', 'buy_online', 'spo
nsored_listings', 'location'])
```

check for the drop down container

In [6]:
```python
result_json['listings']

```

```
      'condition_history': {'ownerCount': 2,
       'accidentCount': 0,
       'recallCount': 3,
       'maintenanceCount': 0,
       'isFleetCar': False,
       'isReportFree': True,
       'titleInfo': {'isSalvage': False,
        'isLemon': False,
        'isTheftRecovered': False,
        'isCleanTitle': True,
        'isFrameDamaged': False},
       'isRentalCar': False,
       'reportPullDate': 1661744254288,
       'isFrameDamaged': False,
       'auto_check_url': 'https://www.truecar.com/abp/api/vendor/autocheck/reports/ic_ux91K1dMJbmoze1sb33MXq7FvaLf9a4nX6TKBL
yIEcT_3qq2m-Q',
       'reportPullDateString': '2022-08-29T03:37:34.288+00:00'},
      'truecar_plus_eligible': False,
      'multilocation': True,
```

find out if lenght of the drop down container matches the item displayed on Website

```
In [7]:  len(result_json['listings'])
Out[7]:  30
```

make checkpoint variable

```
In [8]:  res_item=result_json['listings']
```

Go for the first element in Dictionary

```
In [9]:  result_json['listings'][0]
```

```
 'request_price': None,
 'manufacturer_options': [],
 'bed_length': None,
 'cab_type': None,
 'roof_height': None,
 'options': {},
 'condition_history': {'ownerCount': 2,
  'accidentCount': 1,
  'recallCount': 1,
  'maintenanceCount': 0,
  'isFleetCar': False,
  'isReportFree': False,
  'titleInfo': {'isSalvage': False,
   'isLemon': False,
   'isTheftRecovered': False,
   'isCleanTitle': True,
   'isFrameDamaged': False},
  'isRentalCar': False,
  'reportPullDate': 1660684682850,
  'isFrameDamaged': False,
```

find the relevant key which Column/field we want to fetch

```
In [10]: result_json['listings'][0].keys()

Out[10]: dict_keys(['vehicle', 'dealership', 'payments', 'listed_at', 'distance_retailing', 'images', 'pricing_flags', 'pricing', 'pr
         ice_curve', 'collapse', 'is_fallback_listing'])
```

vehicle information

```
In [11]: result=result_json['listings'][0]['vehicle']
```

Company information

```
In [12]: result['make']
Out[12]: 'Ford'
```

Model information

```
In [13]: result['model']
Out[13]: 'Fusion'
```

Price information

```
In [14]: result['list_price']
Out[14]: 12076.0
```

Distance_Travelled information

```
In [15]: result['mileage']
Out[15]: 118860
```

Manufecture_Year information

```
In [16]: result['year']
Out[16]: 2014
```

```
In [ ]:
```

```
In [ ]:
```

Collect all data of res_item(no. of items in the webpage)

```
In [17]:
brand = []
model = []
mileage = []
year = []
price = []


for result in res_item:

    # brand
    brand.append(result['vehicle']['make'])

    # model
    model.append(result['vehicle']['model'])

    # mileage
    mileage.append(result['vehicle']['mileage'])

    # year
    year.append(result['vehicle']['year'])

    # price
    price.append(result['vehicle']['list_price'])

```

make a pandas DataFrame to print the Records form first Page

In [18]:

```python
df1=pd.DataFrame({'Brand':brand,'Model':model,'Mileage':mileage,'Year':year,'Price':price})
df1
```

Out[18]:

	Brand	Model	Mileage	Year	Price
0	Ford	Fusion	118860	2014	12076.0
1	Ford	Edge	90240	2018	16000.0
2	Ford	Fusion	31590	2016	18998.0
3	Jeep	Wrangler	127650	2012	16990.0
4	Jeep	Patriot	69288	2014	15998.0
5	Kia	Forte	84136	2017	15998.0
6	Nissan	Sentra	43429	2019	19998.0
7	Acura	MDX	106347	2012	18998.0
8	Jeep	Grand Cherokee	98944	2017	19998.0
9	Volvo	XC60	139151	2017	15572.0
10	Ford	Expedition	127042	2015	18998.0
11	Hyundai	Sonata	120171	2016	14998.0
12	Kia	Forte	97607	2015	11649.0
13	Nissan	Sentra	53108	2019	19998.0
14	Land Rover	LR4	116850	2012	16490.0
15	Toyota	Camry	97203	2016	18998.0
16	Mazda	CX-5	122597	2019	19998.0
17	Kia	Soul	91890	2018	14990.0
18	Nissan	Rogue	109047	2018	18998.0
19	Hyundai	Elantra	95237	2019	16998.0
20	Ford	Fusion	50109	2018	19998.0
21	BMW	3 Series	64643	2016	19998.0
22	Chevrolet	Volt	82872	2012	16998.0
23	Honda	Civic	91820	2018	19998.0
24	Kia	Forte	24371	2018	18998.0
25	Ford	Escape	55552	2017	18998.0
26	Ford	Fusion	42564	2014	17998.0
27	Nissan	Altima	81665	2015	15998.0
28	Ford	Fiesta	112349	2015	10998.0
29	Dodge	Avenger	88131	2013	12998.0

now we want to fetch further data from other pages as well i.e from page 2 to 50 and appent them in lists

In [19]:

```python
brand = []
model = []
mileage = []
year = []
price = []


for i in range(2,50):
    headers = {
        'authority': 'www.truecar.com',
        'accept': 'application/json, text/plain, */*',
        'accept-language': 'en-IN,en;q=0.7',
        # Requests sorts cookies= alphabetically
        # 'cookie': f"flag-abt-tcdc-about-us-page=true; flag-abt-hyundai-ioniq-5-experiment=true; flag-abt-true-car-plus
        'if-none-match': 'W/"52db36c6f7181086af53306513cac330"',
        'referer': 'https://www.truecar.com/used-cars-for-sale/listings/price-10000-20000/location-toronto-ks/?page=2',
        'sec-fetch-dest': 'empty',
        'sec-fetch-mode': 'cors',
        'sec-fetch-site': 'same-origin',
        'sec-gpc': '1',
        'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/105.0.0.0
    }

    # response = requests.get('https://www.truecar.com/abp/api/vehicles/used/listings?city=toronto&collapse=true&fallbac
    URL='https://www.truecar.com/abp/api/vehicles/used/listings?city=toronto&collapse=true&fallback=true&include_incenti
#     print(URL)
    response = requests.get(URL, headers=headers)

    result_json = response.json()
    result_item=result_json['listings']

    for result in result_item:

        # brand
        brand.append(result['vehicle']['make'])

        # model
        model.append(result['vehicle']['model'])

        # mileage
        mileage.append(result['vehicle']['mileage'])

        # year
        year.append(result['vehicle']['year'])

        # price
        price.append(result['vehicle']['list_price'])
```

Now printing all of the data fetched from the 1 to 50 pages in a DataFrame

In [20]:

```python
df2=pd.DataFrame({'Brand':brand,'Model':model,'Mileage':mileage,'Year':year,'Price':price})
df2
```

Out[20]:

	Brand	Model	Mileage	Year	Price
0	Nissan	Armada	121490	2015	19998.0
1	Hyundai	Sonata	122443	2012	13599.0
2	Nissan	Sentra	44249	2019	19998.0
3	Jeep	Grand Cherokee	88649	2014	18990.0
4	Toyota	Corolla	51460	2017	19998.0

...	...	...	...	...	...
1435	Hyundai	Santa Fe Sport	58828	2018	19999.0
1436	Kia	Forte	108958	2020	18998.0
1437	Nissan	Rogue	104231	2016	17998.0
1438	Kia	Optima	56541	2016	19998.0
1439	Volkswagen	Jetta	89761	2016	16998.0

1440 rows × 5 columns

checking for individual field

In [21]:
```
brand
```

```
 'Hyundai',
 'Scion',
 'Lexus',
 'Ford',
 'Toyota',
 'Ford',
 'Kia',
 'Kia',
 'Volvo',
 'Jeep',
 'Toyota',
 'Honda',
 'Chevrolet',
 'Ford',
 'Chevrolet',
 'Dodge',
 'Chevrolet',
 'Toyota',
 'Kia',
 'Ford',
```

In [ ]:
```

```

In [ ]:
```

```

Connect to Database - PostgreSQL

In [22]:
```
# pip install sqlalchemy
# pip install PyMySQL
```

In [23]:
```
# install postgras on comupter - pdadmin is UI based Application to run SQL wueries
```

In [24]:
```
# password : "password"
```

In [25]:
```
# create sqlalchemy engine
# engine = sqlalchemy.create_engine('postgresql://postgres:Scaleop7@localhost:5432')
```