**6.WEB SCRAPPING**

**Introduction:**

- scrapping of data from web(loading data from websites)

- unstructured in html

- covertible into spreadsheets/DB

- Major websites have their APIs for web scrapping

**Scrapper:**

* Extract all the data on particular sites.
* Specific data that a user wants

**Process:**

* URL=>
* HTML code=>
* elements(CSS/JS)=>
* scrapes the required data=>
* saves it in required format(csv,xlsx,json)=>

**Applications:**

* Email Marketing
* Sentiment Analysis
* News Monitoring
* Market Research
* Price Monitoring

**Accessing a static website Libraries:**

* BeautifulSoup
* Requests
* Selenium
* Pandas
* webdriver
* webdriver\_manager

**To install any library:**

**!pip install library**

**!pip install –upgrade library #to upgrade to new version**

**BeautifulSoup:**

* Used to scope data from website
* Package bs4: subpackage beautifulsoup

|  |  |  |
| --- | --- | --- |
| Function | Purpose | Attributes |
| BeautifulSoup() | To extract html code from a web page | .texthtml |
| find() | To find first element of a kind | (‘element\_name) |
| find\_all() | To find all elements of a kind | (‘element\_name) |

**Request:**

* Used to send request to a web page
* get(‘URL’)

**Selenium:**

* used to scrape data from static website
* subpackage:webdriver

|  |  |  |
| --- | --- | --- |
| Function | Purpose | Attributes |
| ChromeOptions() | Create an instance of chrome |  |
| .get | Access a webpage | ‘url’ |
| Find\_element | To find element of a kind | By.ID  By.XPATH |
| .click() | To click a button in webpage |  |

XPATH:

//span[text() = ' '] , .//span[@class = ' ']

**Pandas:**

* To save the information in csv, xls, txt files.

**Webdriver:**

* A WebDriver is a browser automation framework.
* It accepts commands and sends them to a browser.

**Webdriver\_manager:**

* WebDriverManager is an open-source Java library that carries out the management (i.e., download, setup, and maintenance) of the drivers required by Selenium WebDriver (e.g., chromedriver, geckodriver, msedgedriver, etc.) in a fully automated manner.

**Project-1:**

webscrapping HTML code of any website

1. import libraries
2. save the url
3. using requests.get() #access the website
4. using BeautifulSoup() #access the HTML code

**Code:**

from bs4 import BeautifulSoup

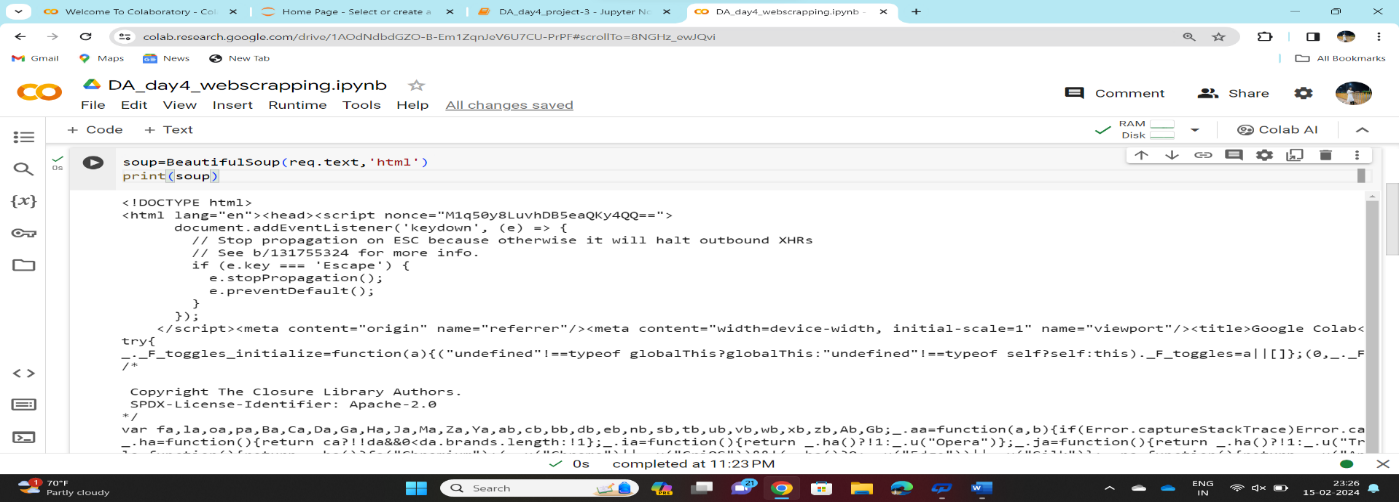
import requests

url="https://colab.research.google.com/drive/1AOdNdbdGZO-B-Em1ZqnJeV6U7CU-PrPF#scrollTo=J\_9QpxwpIOAP"

req=requests.get(url)

soup=BeautifulSoup(req.text,'html')

print(soup)

**Output: **

**Project-2:**

Extracting a table from HTML code of any website and saving it as .csv file

1. import libraries
2. save the url
3. using requests.get()
4. using BeautifulSoup()
5. using find, access the table
6. using find\_all , access the rows of the table
7. using .text, extract only the text(removing html tags)
8. create a dataframe using pandas
9. push all the extracted data into dataframe
10. using to\_csv , save the dataframe in.csv format

**Code:**

from bs4 import BeautifulSoup

import requests

link="https://www.forbesindia.com/article/explainers/top-10-richest-people-india/85909/1"

reque=requests.get(link)

so=BeautifulSoup(reque.text,'html')

print(so)

table=so.find('table')

print(table)

tab=table.find\_all('th')

print(tab)

head=[i.text for i in tab]

print(head)

import pandas as pd

df=pd.DataFrame(columns=head)

print(df)

tr=table.find\_all('tr')

print(tr)

a=0

for i in tr[1:9]:

  er=i.find\_all('td')

  eachrow=[i.text for i in er]

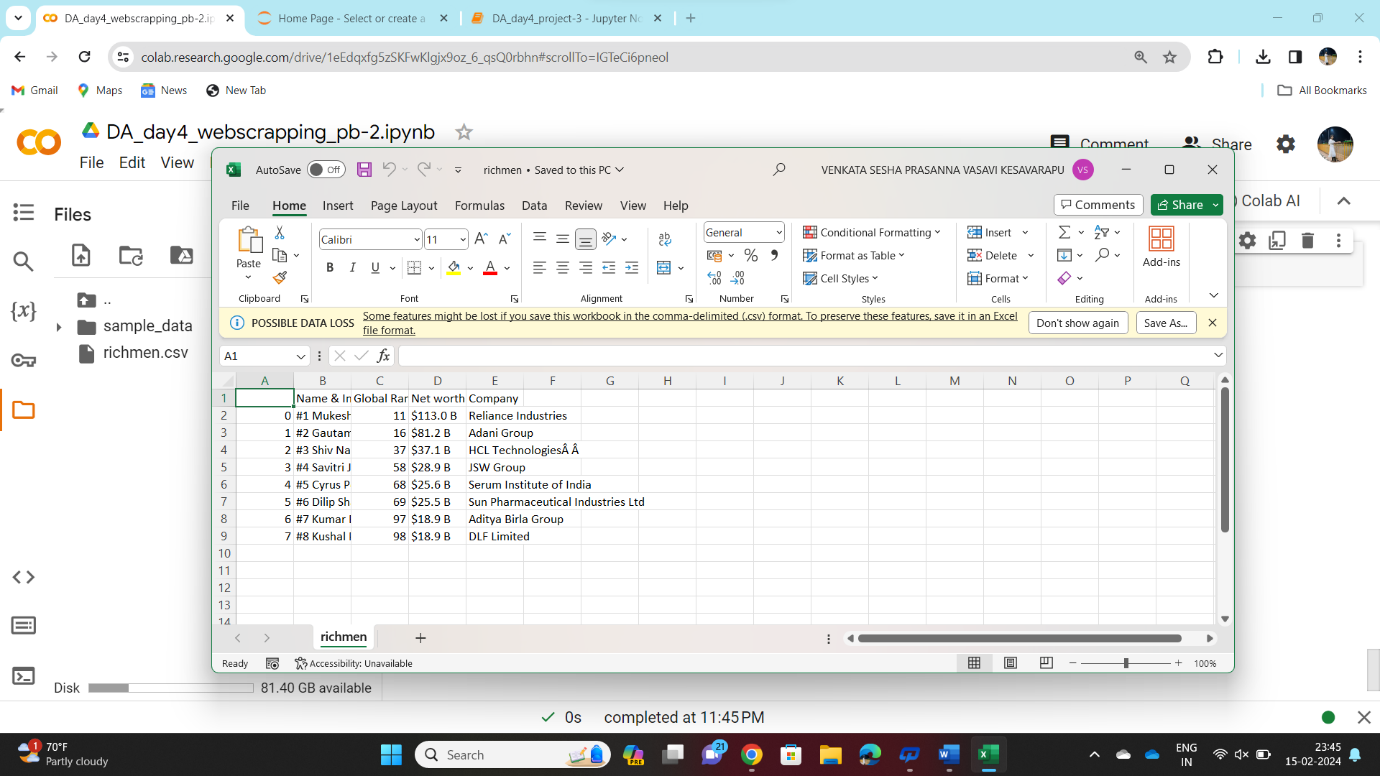
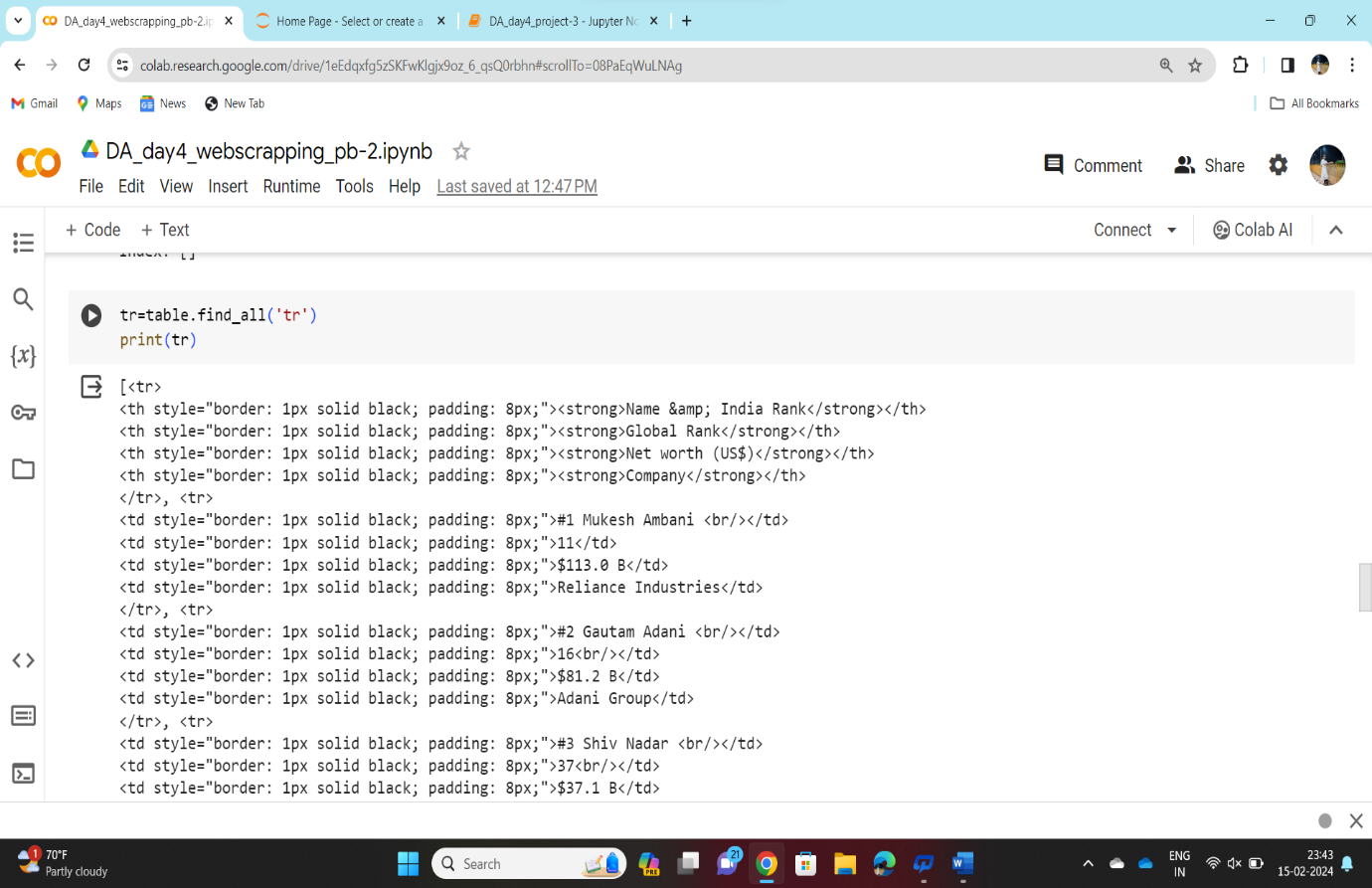
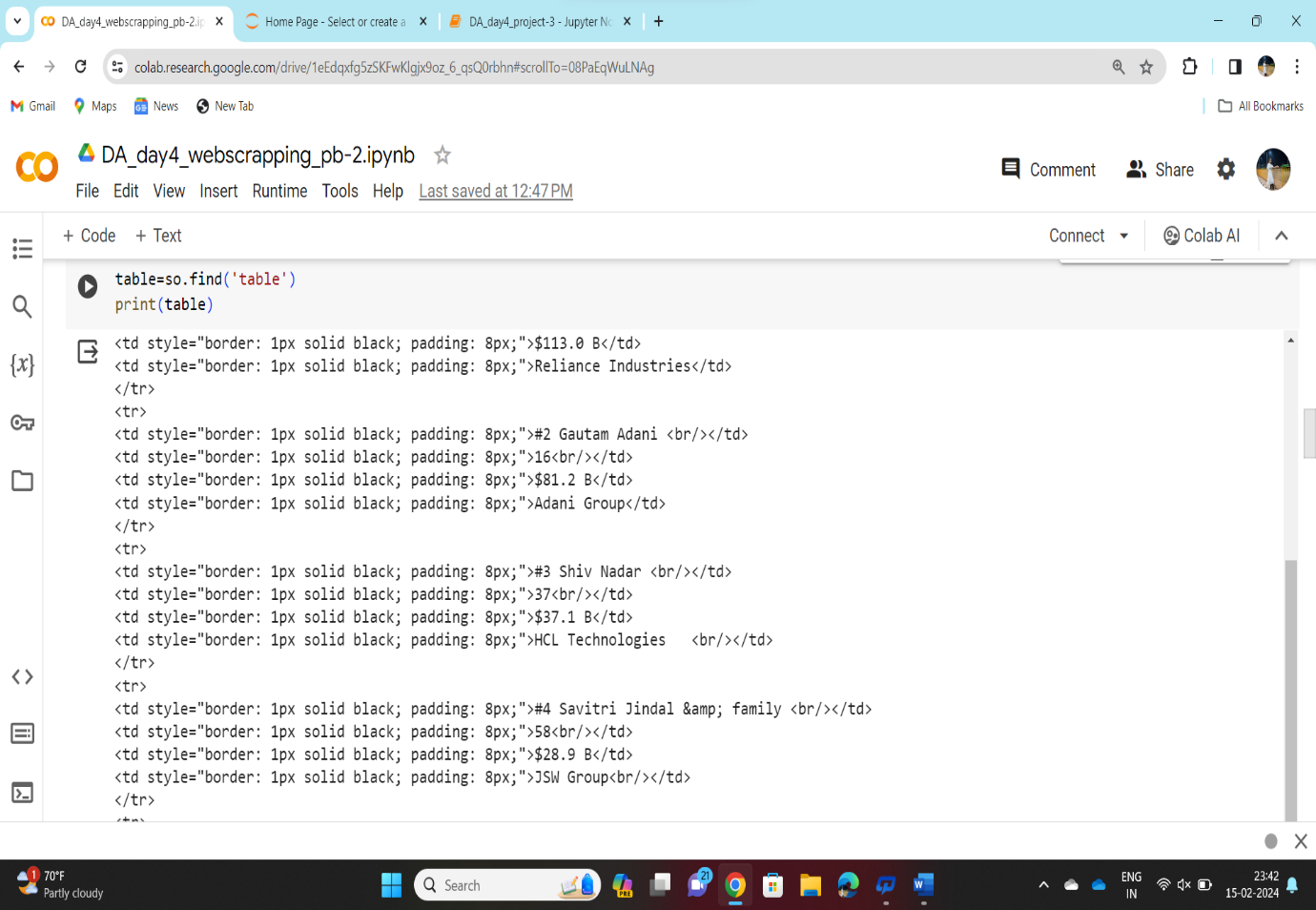
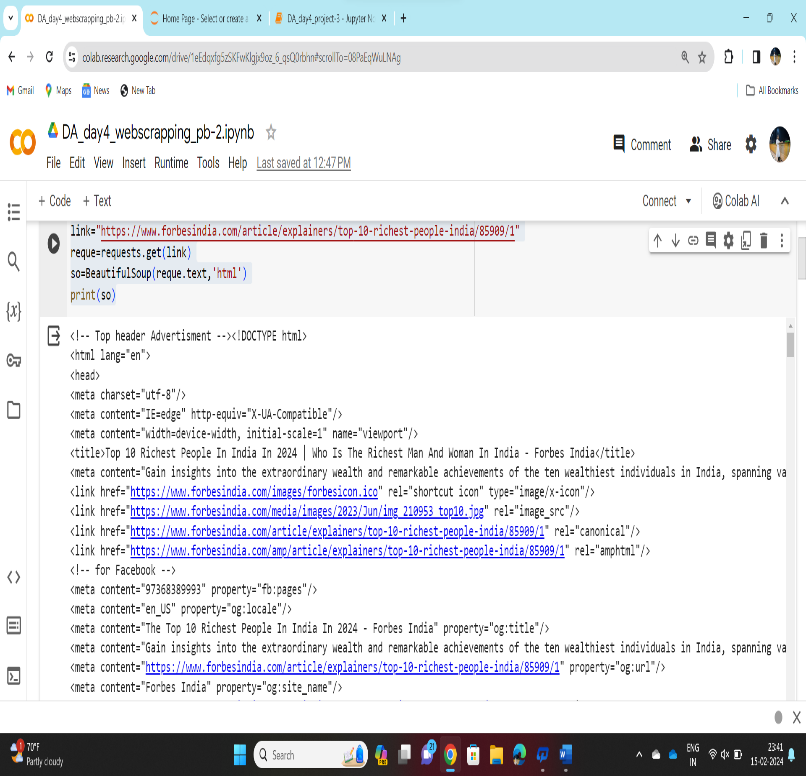
  df.loc[a]=eachrow

  a=a+1

df

df.to\_csv("richmen.csv")  #to is used to save

**Output:**



**Accessing elements**

* Go towebpage
* Right click 🡪 inspect
* Click on
* Now hover on the element you want to access
* The html tags/code for that specific element is highlighted
* Extract the element name and import in your program using XPATH or by ID

**Project-3:**

Extracting dell laptops data from amazon.in website and saving it as .csv name of the laptop, price, no.of reviews.

1. Installing libraries- selenium,webdriver\_manager
2. Import libraries
3. Create a instance of chrome and launch chrome
4. Got to [www.amazon.in](http://www.amazon.in) webpage

**Code:**

!pip install selenium

!pip install webdriver\_manager

import pandas as pd

from selenium import webdriver

from selenium.webdriver.chrome.options import Options

from webdriver\_manager.chrome import ChromeDriverManager

from selenium.webdriver.common.by import By

from selenium.webdriver.common.keys import Keys

#define options and set browser capabilities

options=webdriver.ChromeOptions()

options.add\_argument('--some-option')

#create webdriver instance with options

driver=webdriver.Chrome(options=options)

#access browser capabilities

browser\_name=options.to\_capabilities()["browserName"]

print(browser\_name)

driver.get("https://www.amazon.in/")

search=driver.find\_element(By.ID,"twotabsearchtextbox")

search.send\_keys("Dell Laptops")

driver.find\_element(By.ID,"nav-search-submit-button").click()

driver.find\_element(By.XPATH,"//span[text()='Dell']").click()

names=driver.find\_elements(By.XPATH,".//span[@class='a-size-medium a-color-base a-text-normal']")

L\_names=[i.text for i in names]

print(L\_names)

print(len(L\_names))

prices=driver.find\_elements(By.XPATH,".//span[@class='a-price-whole']")

L\_prices=[i.text for i in prices]

print(L\_prices)

print(len(L\_prices))

L\_prices.pop(0)

L\_prices.pop(0)

L\_prices.pop(0)

print(L\_prices)

print(len(L\_prices))

reviews=driver.find\_elements(By.XPATH,".//span[@class='a-size-base s-underline-text']")

reviews=[i.text for i in reviews]

print(reviews)

print(len(reviews))

headings=["NAME OF THE LAPTOPS","PRICES","REVIEWS"]

df=pd.DataFrame(columns=headings)

print(df)

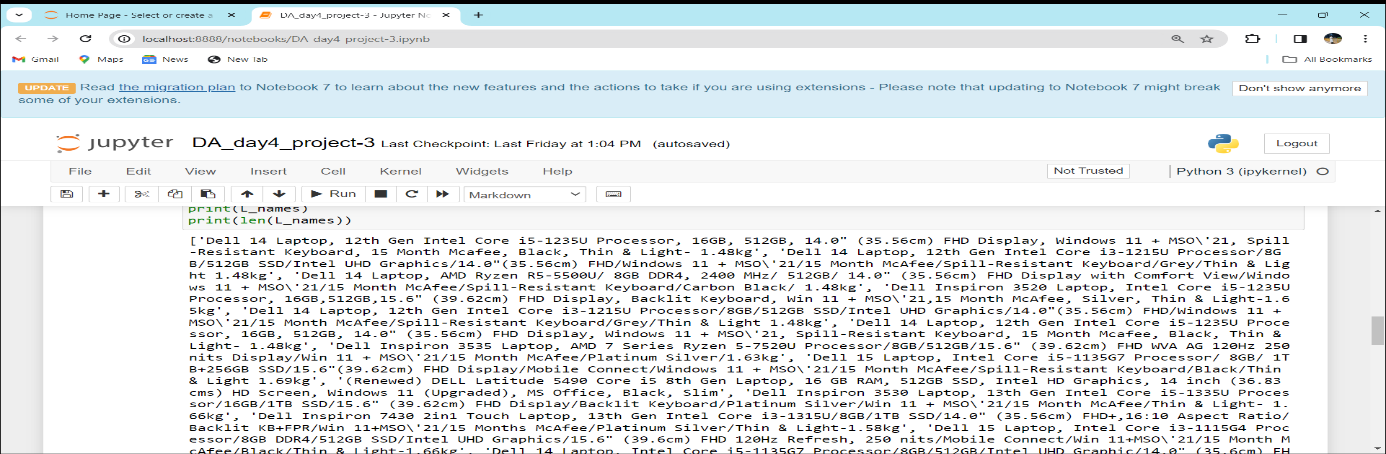
df['NAME OF THE LAPTOPS']=L\_names

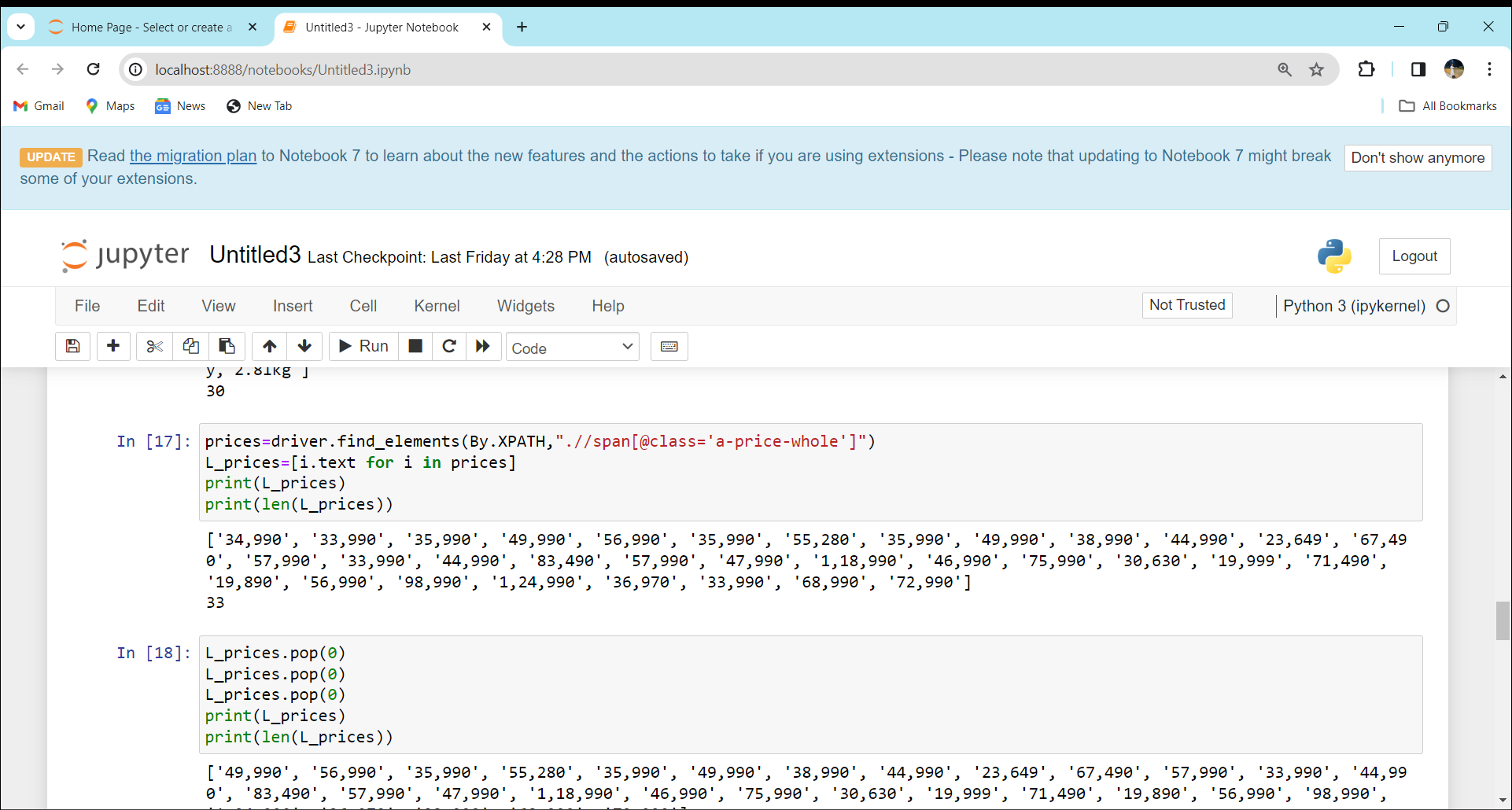
df['PRICES']=L\_prices

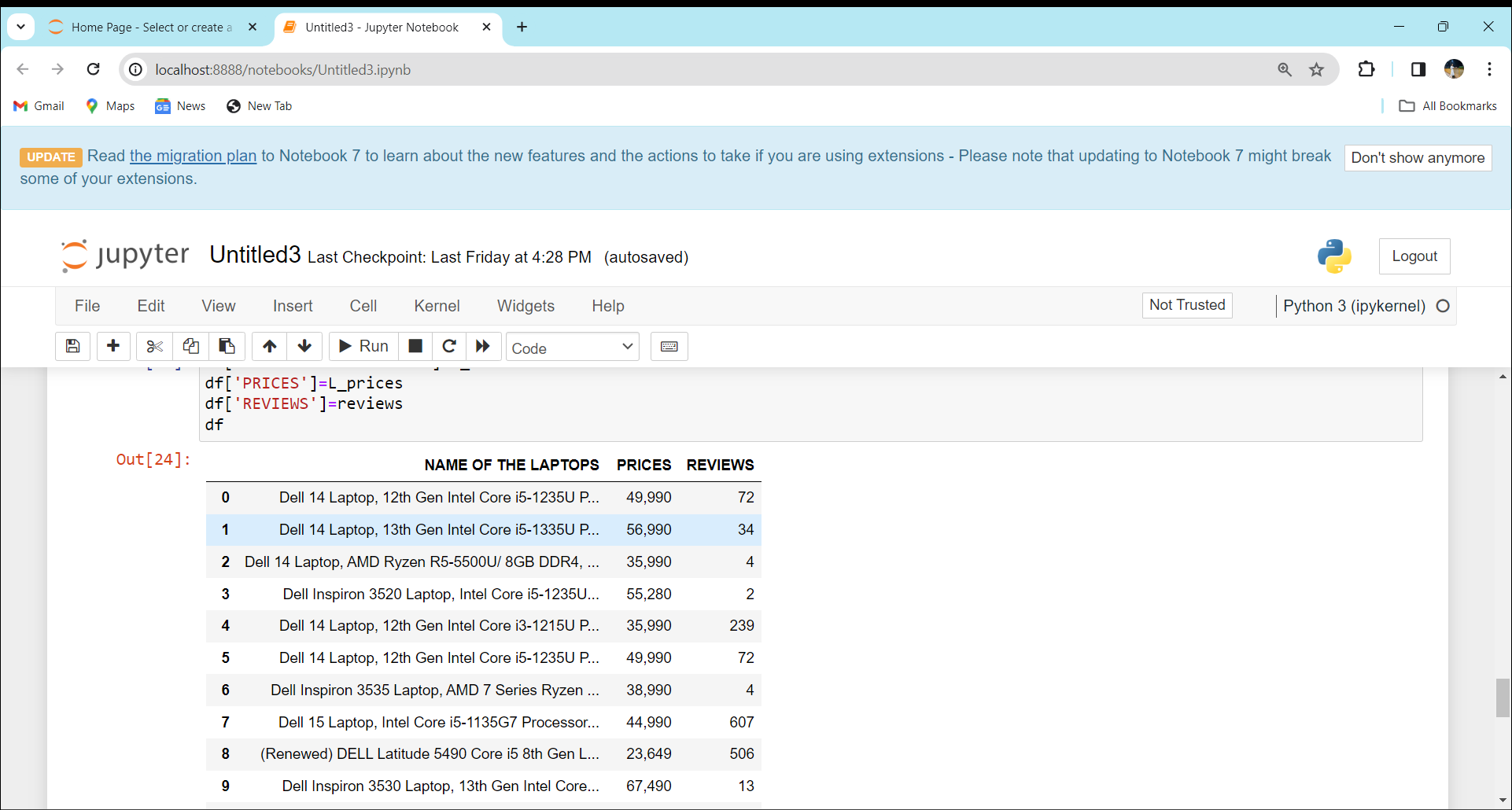
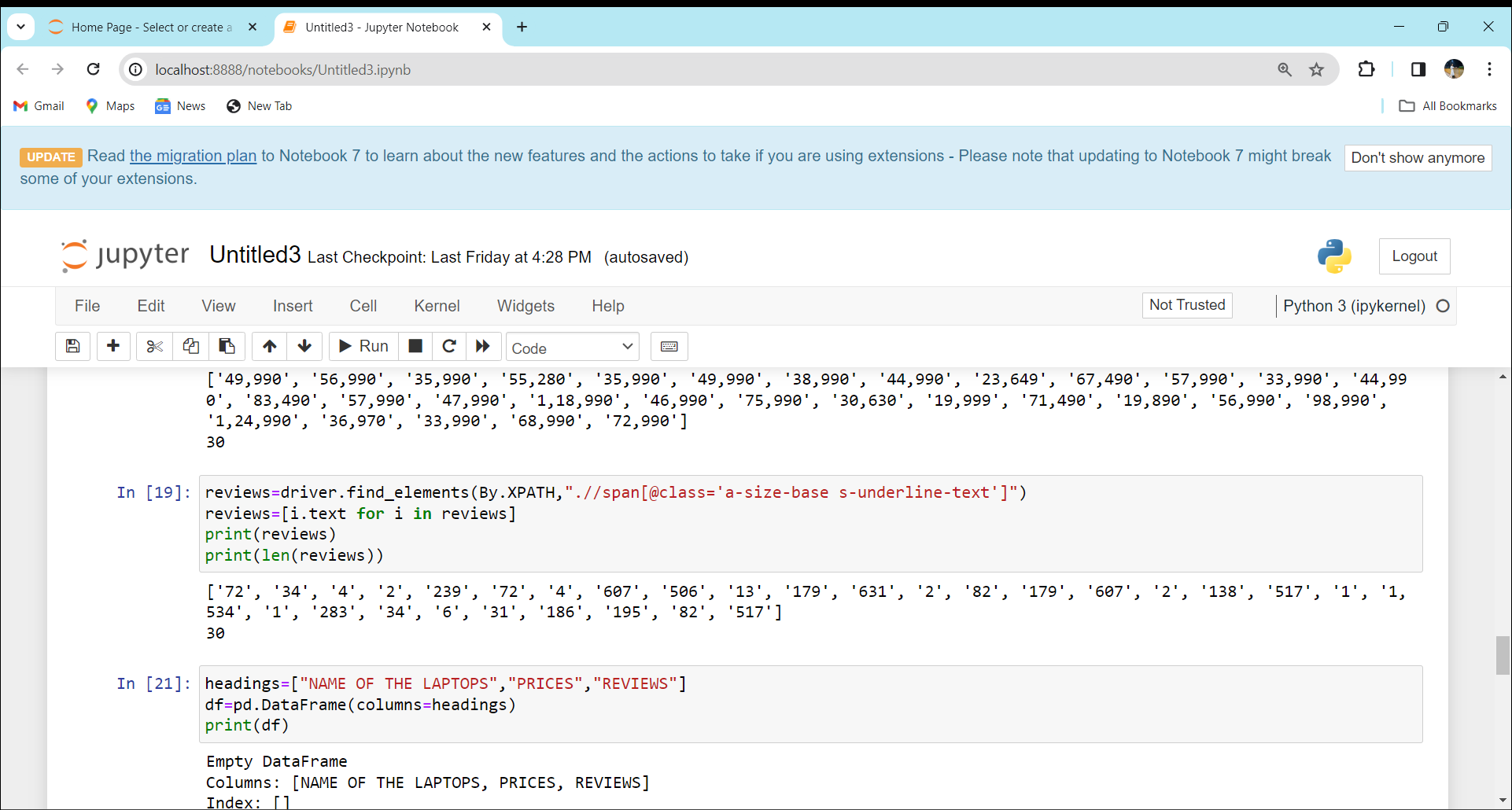
df['REVIEWS']=reviews

df

df.to\_csv("dell\_data.csv")

**Output: **

****

****

