**DEAL SCRAPER**

##### **MINOR PROJECT REPORT**

###### **Submitted by**

##### **TEAM PMA**

**PURU SHARMA RA1511003010404**

##### **ABHISHEK CHAUHAN RA1511003010381**

***For the course***

**15IT322E – Python Programming**

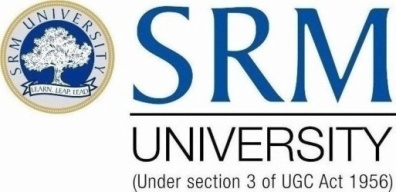
***in partial fulfillment for the award of the degree***

***of***

**Bachelor of Technology**

***in***

**COMPUTER SCIENCE AND ENGINEERING**



**SRM UNIVERSITY**

KATTANKULATHUR

##### **OCTOBER 2017**

ii

**SRM UNIVERSITY**

KATTANKULATHUR

**BONAFIDE CERTIFICATE**

Certified that this project report of **“DEAL SCRAPER”** is the bonafide work of **PURU SHARMA (RA1511003010404)** and **ABHISHEK CHAUHAN (RA1511003010381)** of **Computer Science and Engineering, B.Tech** degree course.They carried out the project work as part of their course **15IT322E -Python Programming.**

**SIGNATURE SIGNATURE**

**Course Instructor Dr. B. Amutha**

**HEAD OF THE DEPARTMENT Computer Science & Engg**

**INTERNAL EXAMINER**

**ABSTRACT**

Our aim was to create a light weight application or script in the **Python Programming Language** which would take an item name from the user and fetch the price of that item from the three popular e-cmmerce websites in India: **Flipkart, Amazon and Snapdeal**. Based on the search query provided by the user, the app will find the most relevant results on these websites and make them available to the user.

Our final output will have:

1. The name of website
2. The price of the most relevant product available on that website
3. A direct link to the product
4. A link to the search query’s results on the website itself and
5. An indicator of the best deal available

The code relies on the individual relevancy check of each website so as to avoid any errors caused by exclusivity in terms of products available. The user can click on the direct product link to buy the item deemed most relevant or can click on the query results to peruse through the options available on each website themselves.

**INTRODUCTION**

With e-commerce having soundly replaced shops as the preferred way to shop for the young generations and the large competition now present in this market, it is necessary when buying a product online that we get the very best deal. Usually when a person is interested in purchasing some product, he has to scroll and search for the same product on all the different websites so as to save as much money as possible. This results in a lot of wasted time. We wanted to save this precious time for the online shoppers in India.

For our Python Project, we wished to use everything that we had learned in class. Having already studied many of the old programming languages before, new concepts such as web scraping interested us the most.

Thus, when we set out to do our project our aim was to use the basic knowledge of these topics gained by us during class and learned about applications and further extensions of these available online. An example of this would be starting from **urllib**, and gradually studying further and learning about **requests**.

**LIBRARIES**

We wanted to employ the concepts that we had studied during class for this project so even though APIs for the popular websites, we manually fetched all the data from them. The libraries we ended up using were either the ones taught in class or largely inspired by them.

They were:

1. **Requests**

To fetch the data from the websites.

1. **BeautifulSoup**

To parse the fetched web-pages and to find the required data within them.

1. **Urllib**

For specific parsing of URLs in certain situations.

1. **Tkinter**

To design the GUI for the application.

1. **webbrowser**

To connect the URLs provided to your web browser of choice.

**RELATED WORK**

There are many websites which do similar work to our application by listing the prices of products from different websites.

Pricedekho, Pricetree, Shopping.com are some examples.

These websites use the APIs provided by the shopping sites to fetch the product listings and store them on their own servers, updating the information in regular intervals. Some of these websites also provide APIs of their own for others to use to compare the prices directly from their data.

Also to be noted is that these websites all have different monetization methods which might made the data unreliable in some cases, whether it be through simple advertising or through affiliate links.

Using these third party APIs was the easiest way to make our project but we chose not to do it because its ease didn’t allow us to learn what we wanted to learn when we decided to make this project.

Instead of using the APIs we fetched the webpages directly from the original shopping sites and scraped the data ourselves.

**INDIVIDUAL CONTRIBUTIONS**

**Puru Sharma (RA1511003010404)** –

Fetched the webpages from the websites and extracted the price and direct hyperlinks from them. Updated the UI to be user-friendly, providing direct hyperlinks for the products and the queries.

**Abhishek Chauhan (RA1511003010381)** –

Designed the UI using Tkinter and scraped the required elements from the Snapdeal data. Added the different exception handling clauses to improve the general stability of the code.

**ALGORITHM/METHOD**

1. We first take the search query from the user in a UI created using **Tkinter.** We then create specific URLs for the query for each of the three websites’s search engines.
2. Now, we can fetch the webpage from these URLs using **Requests.**
3. Using **BeautifulSoup,** we can now parse these HTML pages.
4. By finding the specific divs and classes within them which contain the price and the link to the most relevant product, we can fetch these links.
5. This information is cleaned up using functions with **urllib.**
6. We finally provide this information to the user within the GUI.
7. The links are made hyperlinks within the ListBox using **webbrowser.**

**PROGRAM**

Running the code will open the Tkinter interface with an entry box to enter the search query. On entering the same and clicking the Search button, the rest of the code is executed.

**Code:**

|  |
| --- |
| **import** requests **from** bs4 **import** BeautifulSoup **import** urllib **from** tkinter **import** \* **import** webbrowser  *#to convert the query into url friendly format*  **def** remove\_whitespace(x):  **try**:  x = x.replace(**" "**, **"+"**)  **except**:  **pass  return** x  *#to take input from user and return it in a search query*  **def** TakeQuery():  product\_input = svalue.get()  product\_input = remove\_whitespace(product\_input)  **return** product\_input  *#for amazon headers are needed*  headers = {**'User-Agent'**: **'Mozilla/5.0 (Windows NT 6.3; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/54.0.2840.71 Safari/537.36'**} |

**def** Searching():  
 product\_name = TakeQuery();  
  
 *#amazon* **try**:  
 search\_query\_amazon=**"https://www.amazon.in/s/url=search-alias%3Daps&field-keywords="** + product\_name  
 r1=requests.get(search\_query\_amazon, headers=headers)  
 soup\_amazon = BeautifulSoup(r1.content, **'html.parser'**)  
  
 *# This is fetching how much you save  
 #amazonPrice = soup\_amazon.findAll('span', class\_='a-color-price')[0].contents[1].replace(',', '')  
  
 #Getting the amazon url* amazonLink = soup\_amazon.findAll(**'a'**, class\_=**'a-link-normal'**)[0]  
  
 amazonPrice = soup\_amazon.findAll(**'span'**, class\_=**'a-price-whole'**)  
 **if**(len(amazonPrice) > 0):  
 amazonPrice = amazonPrice[0].string.replace(**','**, **''**)  
 **else**:  
 *# fallback code for now, but shows how much you save. Most likely not required* amazonPrice = soup\_amazon.findAll(**'span'**, class\_=**'a-color-price'**)[0].contents[1].replace(**','**, **''**)  
 amazonPrice=int(amazonPrice);  
  
 amazonUrl = amazonLink[**'href'**]  
 *# Get the parameters to redirect to from search url* **for** amzu **in** amazonUrl.split(**'&'**):  
 *# Look for url to redirect to* **if** amzu.find(**'url='**) > -1:  
 amazonUrl = amzu.split(**'='**)[1]  
 **break** *# Decode the encoded string and remove the additional information* amazonUrl = urllib.parse.unquote(urllib.parse.unquote(amazonUrl)).split(**'?'**)[0]  
 **except**:  
 amazonPrice = **'Product Unavailable'** search\_query\_amazon = **'----------'** amazonUrl = **'----------'**

*#flipkart***try**:  
 search\_query\_flipkart=**"https://www.flipkart.com/search?q="** + product\_name  
 r2=requests.get(search\_query\_flipkart);  
 soup\_flipkart = BeautifulSoup(r2.content, **'html.parser'**)  
  
 flipkartPrice = soup\_flipkart.findAll(**'div'**, class\_=**'\_1vC4OE'**)[0].contents[-2].replace(**','**,**''**)  
 flipkartPrice = int(flipkartPrice)  
  
 flipkartUrl = soup\_flipkart.find(**'div'**, class\_=**'\_1vC4OE'**).parent.parent[**'href'**]  
 flipkartUrl = **'https://www.flipkart.com'** + flipkartUrl.split(**'?'**)[0]  
**except**:  
 flipkartPrice = **'Product Unavailable'** search\_query\_flipkart = **'----------'** flipkartUrl = **'----------'***#snapdeal***try**:  
 search\_query\_snapdeal=**"https://www.snapdeal.com/search?keyword="**+product\_name;  
 r3=requests.get(search\_query\_snapdeal);  
 soup\_snapdeal = BeautifulSoup(r3.content, **'html.parser'**)  
  
 snapdealPrice = soup\_snapdeal.findAll(**'span'**, class\_=**'product-price'**)[0][**'data-price'**]  
 snapdealPrice = int(snapdealPrice)  
*# snapdealUrl = soup\_snapdeal.find('span', {'class': 'product-desc-rating'}).find('span', {'class':['dp-widget-link', 'noUdLine', 'hashAdded']}).findAll('a')[0]['href']* snapdealUrl = soup\_snapdeal.find(**'span'**, class\_=**'product-price'**).parent.parent.parent[**'href'**]  
**except**:  
 snapdealPrice = **'Product Unavailable'** search\_query\_snapdeal = **'----------'** snapdealUrl = **'----------'**

amazonTrigger=0  
flipkartTrigger=0  
snapdealTrigger=0

**if** (amazonPrice <= flipkartPrice)**and**(amazonPrice <= snapdealPrice):  
 amazonTrigger=1  
**if** (flipkartPrice <= amazonPrice)**and**(flipkartPrice <= snapdealPrice):  
 flipkartTrigger=1  
**if** (snapdealPrice <= amazonPrice)**and**(snapdealPrice <= flipkartPrice):  
 snapdealTrigger=1  
  
*# -------------------TKINTER-----------------------*Lb1.delete(0, END)  
LbPrice.delete(0, END)  
LbLink.delete(0, END)  
LbQuery.delete(0, END)  
LbDeal.delete(0, END)  
Lb1.insert(1, **'WebSite'**)  
Lb1.insert(2, **' '**)  
Lb1.insert(3, **'Amazon '**)  
Lb1.insert(4, **'Flipkart'**)  
Lb1.insert(5, **'Snapdeal'**)

LbPrice.insert(1, **'Price ( ₹ )'**)  
LbPrice.insert(2, **' '**)  
LbPrice.insert(3, amazonPrice)  
LbPrice.insert(4, flipkartPrice)  
LbPrice.insert(5, snapdealPrice)  
LbLink.insert(1, **'Direct Link'**)  
LbLink.insert(2, **' '**)  
LbLink.insert(3, amazonUrl)  
LbLink.insert(4, flipkartUrl)  
LbLink.insert(5, snapdealUrl)  
LbQuery.insert(1, **'Your Query Results'**)  
LbQuery.insert(2, **' '**)  
LbQuery.insert(3, search\_query\_amazon)  
LbQuery.insert(4, search\_query\_flipkart)  
LbQuery.insert(5, search\_query\_snapdeal)  
LbDeal.insert(1, **' '**)  
LbDeal.insert(2, **' '**)

**if** (amazonTrigger == 1):  
 LbDeal.insert(3, **'BEST DEAL!'**)  
**else**:  
 LbDeal.insert(3, **' '**)  
**if** (flipkartTrigger == 1):  
 LbDeal.insert(4, **'BEST DEAL!'**)

**else**:  
 LbDeal.insert(4, **' '**)  
 **if** (snapdealTrigger == 1):  
 LbDeal.insert(5, **'BEST DEAL!'**)  
 **else**:  
 LbDeal.insert(5, **' '**)  
 Lb1.pack(side=LEFT)  
 LbPrice.pack(side=LEFT)  
 LbLink.pack(side=LEFT)  
 LbQuery.pack(side=LEFT)  
 LbDeal.pack(side=LEFT)

root = Tk(className=**"dealScraper"**)  
root.geometry(**"1400x150"**)  
svalue = StringVar() *# defines the widget state as string*w = Entry(root, textvariable=svalue, width=70) *# adds a textarea widget*w.pack()  
Lb1 = Listbox(root, width=20)  
LbPrice = Listbox(root, width=20)  
LbLink = Listbox(root,width=80)  
LbQuery = Listbox(root, width=90)  
LbDeal = Listbox(root, width=20)

**def** internet(event):  
 weblink = LbLink.get(ACTIVE)  
 webbrowser.open(weblink)

**def** internet2(event):  
 weblink = LbQuery.get(ACTIVE)  
 webbrowser.open(weblink)  
LbLink.bind( **"<Double-Button-1>"** , internet)  
LbQuery.bind( **"<Double-Button-1>"** , internet2)  
Enter = Button(root, text=**"Search"**, command=Searching,width=30,height=2)

Enter.pack()

root.mainloop()

**RESULTS**

**CONCLUSION**

This project provides a simple way to look up anything you wish to buy online in India and searches for the best deal available at the time on the major e-commerce websites that operate here. Our script is simple, light weight and can be modified for other websites and search engines.

However when we started work on this project, this was only our secondary aim.

Our primary aim was to learn how web scraping is done. It was to learn to practically apply what we had studied in class while also delving in more advanced and up to date techniques to do the same.

In trying different techniques to make DealScraper, hitting dead ends, and then trying again, over and over until we finally did it, we succeeded in accomplishing both our goals.

**REFERENCES**

1. BeautifulSoup: <https://pypi.python.org/pypi/beautifulsoup4>
2. Requests: <http://docs.python-requests.org/en/master/>
3. Tkinter: https://docs.python.org/2/library/tkinter.html
4. Webbrowser: https://docs.python.org/2/library/webbrowser.html
5. Urllib: https://docs.python.org/2/library/urllib2.html