**PRODUCT PRICE TRACKER**

*A*

*Mini Project Report*

*Submitted in partial fulfilment of the*

*Requirements for the award of the Degree of*

**BACHELOR OF ENGINEERING**

IN

**INFORMATION TECHNOLOGY**

By

**Shaik Mohammed Sameer 1602-20-737-168**

**Shoaib Aleemuddin 1602-20-737-170**

**Sai Vinayaka Venkata Prateek 1602-20-737-164**

****

**Department of Information Technology**

**Vasavi College of Engineering (Autonomous)**

**(Affiliated to Osmania University)**

**Ibrahimbagh, Hyderabad-31**

**2022**

**Vasavi College of Engineering (Autonomous)**

**(Affiliated to Osmania University)**

**Hyderabad-500 031**

**Department of Information Technology**

****

**DECLARATION BY THE CANDIDATE**

We, **SHAIK MOHAMMED SAMEER**, **SHOAIB ALEEMUDDIN, VENKATA PRATEEK,** bearing hall ticket numbers, **1602-20-737-168, 1602-20-737-170** and **1602-20-737-164**, hereby declare that the project report entitled **“PRODUCT PRICE TRACKER “**is submitted in partial fulfilment of the requirement for the award of the degree of **Bachelor of Engineering** in **Information Technology**

This is a record of bonafide work carried out by us and the results embodied in this project report have not been submitted to any other university or institute for the award of any other degree or diploma.

**Shaik Mohammed Sameer**

**1602-20-737-168**

**Shoaib Aleemuddin**

**1602-20-737-170**

**Venkata Prateek**

**1602-20-737-164**

(Faculty In-Charge) (Head, Dept of IT)

**VASAVI COLLEGE OF ENGINEERING (AUTONOMOUS)**

**(AFFILIATED TO OSMANIA UNIVERSITY)**

**HYDERABAD - 500 031**

**Department of Information** **Technology**

****

**BONAFIDE CERTIFICATE**

This is to certify that the project entitled **“PRODUCT PRICE TRACKER”** being submitted by **SHAIK MOHAMMED SAMEER**, **SHOAIB ALEEMUDDIN, SAI VINAYAKA VENKATA PRATEEK,** bearing hall ticket numbers, **1602-20-737-168, 1602-20-737-170** and **1602-20-737-164**, in partial fulfillment of the requirements for the completion of MINI PROJECT of Bachelor of Engineering in Information Technology is a record of bonafide work carried out by them under my guidance.

Dr. K. Ram Mohan Rao

Faculty I/C HOD, IT

**ACKNOWLEDGEMENT**

We extend our sincere thanks to Dr. S. V. Ramana, Principal, Vasavi College of Engineering for his encouragement.

We express our sincere gratitude to Dr. K. Ram Mohan Rao, Professor & Head, Department of Information Technology, Vasavi College of Engineering, for introducing the Mini-Project module in our curriculum, and also for his suggestions, motivation, and co-operation for the successful completion of our Mini Project.

We also want to thank and convey our gratitude towards our mini project coordinators **Mrs C. Sireesha** and **Mr N. David Raju**, for guiding us in understanding the process of project development & giving us timely suggestions at every phase.

We would also like to sincerely thank the project reviewers for their valuable inputs and suggestions.

**ABSTRACT**

With the rapid advancements in the Electronic Commerce sector, modern consumers are becoming increasingly aware of the prospects of the various pricing strategies, which have made their purchase decisions more sophisticated.

So we made a simple price tracker using python that regularly compares the prices of your favorite stuff on amazon or Flipkart and sends you a customized notification email at your registered email to tell you if the prices fell down below the desired price. This uses python and implements beautiful soup to scrap data from websites, smtplib to send mail, tkinter to make GUI, time and requests modules and libraries.

It helps users to save time as they don't need to manually open the amazon or flipkart and check drop in price every day, they just need to copy the URL of product and add to tracker and input desired price, as soon as price drops user gets notified and can buy product before everyone else does. It would also help to consider factors like product demand, minimum margins, cost within local and international markets, and more.

Instead of manually monitoring your competitors’ pricing and other product-related data, you can use price tracking software to do it automatically. It frees you a significant amount of time you can spend on other productive tasks.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| S.No | Topic | Page No. |
| 1. | Introduction | 7 |
| 2. | Technology | 8 |
| 3. | Design | 9 |
| 4. | Implementation | 14 |
| 5. | Testing | 21 |
| 6. | Results | 23 |
| 7. | Additional Knowledge Acquired | 30 |
| 8. | Conclusion and Future work | 31 |
| 9. | References | 32 |

**INTRODUCTION**

With the rapid advancements in the Electronic Commerce sector, modern consumers are becoming increasingly aware of the prospects of the various pricing strategies, which have made their purchase decisions more sophisticated. Recent literature portrays an increased tendency among online shoppers to track product prices and wait for price markdowns prior to making their purchase decisions. This act of consumers is seen as strategic purchase behavior.

A product price tracker tool is used to track, analyse, and compare product pricing from competitors’ websites. It is also called a price monitoring tool that gathers information like a product’s availability, visual representation, promotions, customer reviews, etc., apart from pricing. These tools offer dynamic pricing analysis that can equip you with automated, robust pricing optimization. They also enable you to predict changes in demand volume associated with a pricing model and help you determine correct pricing.

Price tracking tools are beneficial to companies dealing with products, retailers, e-commerce product-based sites, marketplaces, manufacturers, and even end-users. A price tracker is a technical solution that helps (online) retailers, wholesalers, and manufacturers track prices of competitors and dealers. It makes the process of tracking prices easier and less painful, and it also gives information to make pricing decisions. A price tracking tool provides you with an accurate insight into pricing to formulate a rock-solid pricing strategy. It will help you make informed decisions to keep your price at an optimal range to gain maximum sales and profit margins**.**

This Product Price Tracker ismade using python that regularly compares the prices of your favorite stuff on Amazon or Flipkart and sends you a customized notification email at your registered email to tell you if the prices fell down below the desired price. This uses python and implements beautiful soup to scrap data from websites, smtplib to send mail, tkinter to make GUI, time and requests modules and libraries. It helps users to save time as they don't need to manually open the amazon or flipkart and check drop in price every day, they just need to copy the URL of product and add to tracker and input desired price, as soon as price drops user gets notified through an email.

In email it will give the link of the product, current price and ratings of the product at that time.

Therefore, instead of checking the product details every now and then we can easily track

product data by using this simple tool.

**TECHNOLOGY**

**Hardware Requirements**

* 512 MB RAM
* 2GB HDD
* CORE i3/i5

**Software Requirements**

* Windows XP/ Windows 2000
* PYTHON INTERPRETER

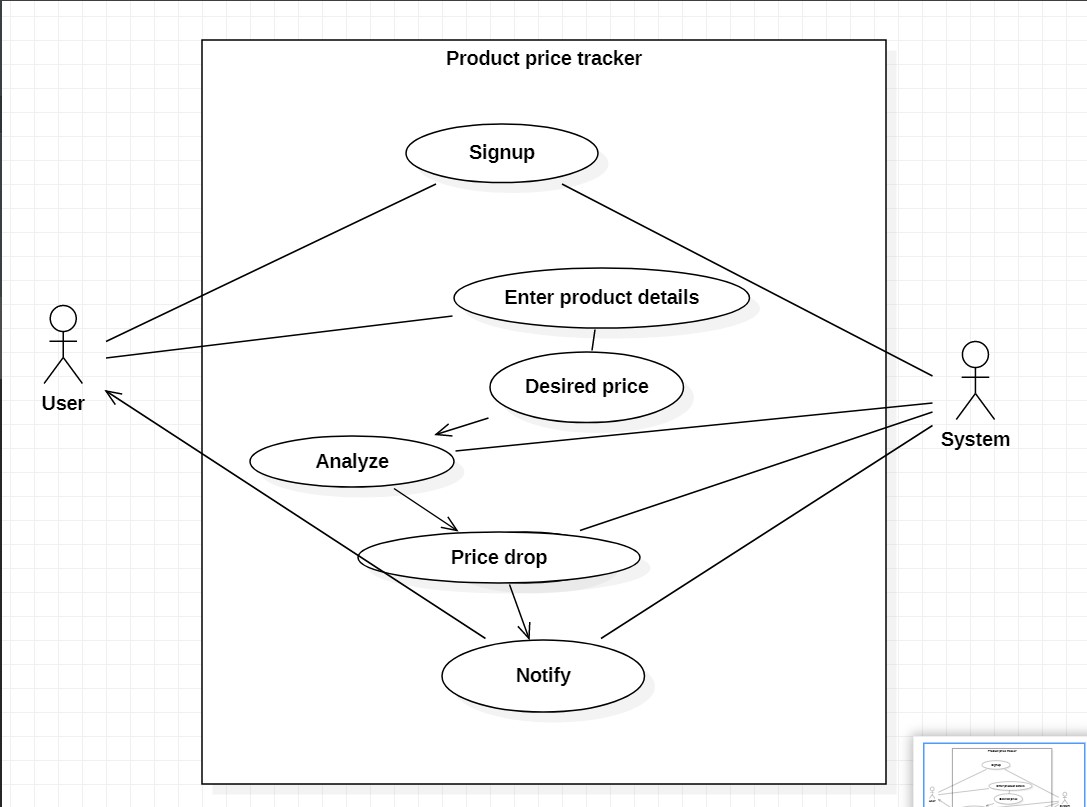
**Packages used**

* Tkinter
* Requests
* Smtplib
* Beautifulsoup

**PROPOSED WORK**

**DESIGN –**

**USE CASE DIAGRAM**



**Use case descriptions**

**1.**

Use case ID: UC01

Name: Email details

Actors: User

Description : Allows the user to enter email and password

Pre-conditions: Enter email and password

Post-conditions: Both email and password must be filled

Main flow:

|  |  |
| --- | --- |
| **User** | **System** |
|  | 1. Displays two textboxes |
| 2.Enter username and password |  |
|  | 3. Allow submit to takes to next page if both email and password is filled |

2.

Use case ID: UC02

Name: E-site selection

Actors: User

Description : Allows the user to select E-site

Pre-conditions: Both email and password should be filled

Post-conditions: An E-site must be selected

Main flow:

|  |  |
| --- | --- |
| **User** | **System** |
|  | 1. Display radio-buttons for selecting E-site |
| 2. User select an E-site |  |
|  | 3. Takes to next respective page |

3.

Use case ID: UC03

Name: Product details

Actors: User

Description : Allows the user to enter product URL and desired price

Pre-conditions: Selection of an E-site

Post-conditions: Both URL and desired price must be filled

Main flow:

|  |  |
| --- | --- |
| **User** | **System** |
|  | 1. Displays two textboxes |
| 2.Enter product URL and desired price |  |
|  | 3. Allow submit to takes to next page if both product URL and desired price is filled |

4.

Use case ID: UC04

Name: Evaluates

Actors: System

Description: System takes the URL and price given by user

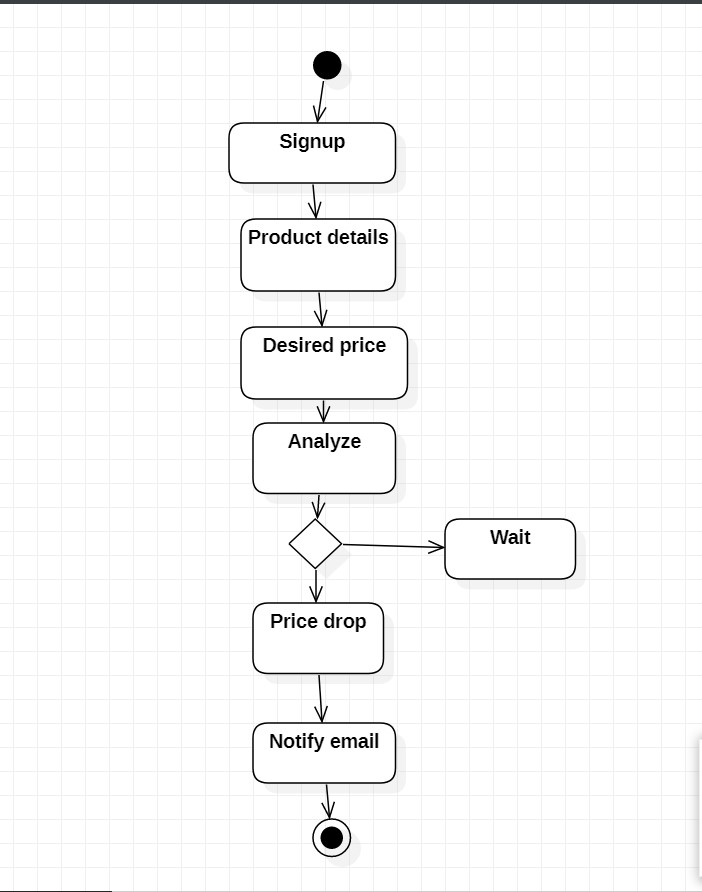
Pre-conditions: User should give his URL and desired price

Post-conditions: displaying message dialog box showing information on whether mail is sent or not

Main Flow:

|  |  |
| --- | --- |
| **User** | **System** |
|  | 1.Evaluates the response given by the user  And display a message box |

**ACTIVITY DIAGRAM**

****

**IMPLEMENTATION**

**Source code –**

from tkinter import \*

from tkinter import messagebox

import requests

import smtplib

from bs4 import BeautifulSoup

def check\_internet():

try:

requests.get("https://google.com")

except Exception:

sho = Tk()

sho.geometry("600x200")

sho.maxsize(600,200)

sho.minsize(600,200)

sho.title("P.P.T.")

Label(sho, text="PRODUCT PRICE TRACKER",

font="comicsansms 13 bold", pady=15).place(relx=0.3, rely=0)

Label(sho, text="Make sure you're connected to the internet!",

font="comicsansms 13 bold", pady=15).place(relx=0.3, rely=0.4)

img = PhotoImage(file='warn.png')

img1 = img.subsample(4, 4)

Label(sho, image=img1).place(height=100, width=100, relx=0.1, rely=0.3)

sho.mainloop()

quit()

def sendmail(product\_name, URL, desired\_price, price, ratings):

'''Function called when the email needs to be sent '''

server = smtplib.SMTP('smtp.gmail.com', 587)

server.ehlo()

server.starttls()

server.ehlo()

server.login(emailvalue.get(), passwrdvalue.get())

subject = f"Price down for {product\_name}"

body = f"The price for '{product\_name}' is now {price} which is in your desired range, {desired\_price}! And rating of product is {ratings}. Visit {URL} for more info."

msg = f"Subject: {subject}\n\n{body}"

msg1 = msg.encode()

server.sendmail(emailvalue.get(), emailvalue.get(), msg1)

messagebox.showinfo("SUCCESSFUL", "Mail has been Sent")

server.quit()

nxt1.destroy()

def amazon():

try:

URL = urlentry.get()

desired\_price = float(priceentry.get())

page = requests.get(URL, headers=headers)

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

product\_name = soup.find(id="productTitle").get\_text().strip()

price = float(soup.find("span", {"class": "a-price a-text-price a-size-medium apexPriceToPay",

"class": "a-offscreen"}).get\_text().replace('₹', '').replace(',', ''))

ratings = soup.find("span", {"class": "a-icon-alt"}).get\_text()

if price <= desired\_price:

sendmail(product\_name, URL, desired\_price, price, ratings)

else:

messagebox.showinfo("INFO", "Price is more than your range ")

nxt1.destroy()

except Exception as e:

messagebox.showwarning("WARNING", "Please enter required details")

def flipkart():

try:

URL = urlentry.get()

desired\_price = float(priceentry.get())

page = requests.get(URL, headers=headers)

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

product\_name = soup.find(

"span", {"class": "B\_NuCI"}).get\_text().strip()

price = float(soup.find("div", {"class": "\_30jeq3 \_16Jk6d"}).get\_text(

).replace(',', '').replace('₹', ''))

ratings = soup.find('div', {"class": "\_3LWZlK"}).get\_text()

if price <= desired\_price:

sendmail(product\_name, URL, desired\_price, price, ratings)

else:

messagebox.showinfo("INFO", "Price is more than your range :(")

nxt1.destroy()

except Exception as e:

messagebox.showwarning("WARNING", "Enter required details")

def next2():

nxt.destroy()

global nxt1

nxt1 = Tk()

nxt1.geometry("500x250")

nxt1.maxsize(500, 250)

nxt1.minsize(500, 250)

nxt1.title("AMAZON PRICE TRACKER")

Label(nxt1, text="Please enter the product url and the desired price",

font="comicsansms 10 italic", pady=15).place(relx=0.1, rely=0.2)

url = Label(nxt1, text="URL")

desired\_price = Label(nxt1, text="Desired Price")

url.place(relx=0.2, rely=0.4)

desired\_price.place(relx=0.2, rely=0.5)

global urlentry, priceentry

urlentry = Entry(nxt1, textvariable=urlVal)

priceentry = Entry(nxt1, textvariable=priceVal)

urlentry.place(relx=0.4, rely=0.4)

priceentry.place(relx=0.4, rely=0.5)

Button(nxt1, text="Submit", background = "gold", command=amazon).place(relx=0.7, rely=0.7)

img = PhotoImage(file='ama.png')

img1 = img.subsample(4, 4)

Label(nxt1, image=img1).place(height=40, width=150, relx=0.3, rely=0.05)

nxt1.mainloop()

def next3():

nxt.destroy()

global nxt1

nxt1 = Tk()

nxt1.geometry("500x250")

nxt1.maxsize(500, 250)

nxt1.minsize(500, 250)

nxt1.title("FLIPKART PRICE TRACKER")

Label(nxt1, text="Please enter the product url and the desired price",

font="comicsansms 10 italic", pady=15).place(relx=0.1, rely=0.2)

url = Label(nxt1, text="URL")

desired\_price = Label(nxt1, text="Desired Price")

url.place(relx=0.2, rely=0.4)

desired\_price.place(relx=0.2, rely=0.5)

global urlentry, priceentry

urlentry = Entry(nxt1, textvariable=urlVal)

priceentry = Entry(nxt1, textvariable=priceVal)

urlentry.place(relx=0.4, rely=0.4)

priceentry.place(relx=0.4, rely=0.5)

Button(nxt1, text="Submit", background = "gold", command=flipkart).place(relx=0.7, rely=0.7)

img = PhotoImage(file='f.png')

img1 = img.subsample(2,2)

Label(nxt1, image=img1).place(height=40, width=150, relx=0.3, rely=0.05)

nxt1.mainloop()

def next1():

if not emailentry.get():

messagebox.showwarning("WARNING", "Please provide email id")

elif not passwrdentry.get():

messagebox.showwarning("WARNING", "Please provide password")

else:

root.destroy()

global nxt

nxt = Tk()

nxt.geometry("1000x250")

nxt.maxsize(1000, 250)

nxt.minsize(1000, 250)

nxt.title("PRODUCT PRICE TRACKER")

img = PhotoImage(file='pt1.png')

img1 = img.subsample(2,2)

Label(nxt, image=img1).place(height=40, width=150, relx=0, rely=0.02)

Label(nxt, text="Welcome to product price tracker!",

font="comicsansms 13 bold", pady=15).place(relx=0.4, rely=0)

Label(nxt, text=" A tool which you can use to track prices for a product on Amazon and Flipkart",

font="comicsansms 13 bold", pady=15).place(relx=0.2, rely=0.2)

Label(nxt, text=" Where to track price?",

font="comicsansms 13 bold", pady=15).place(relx=0.2, rely=0.4)

Radiobutton(nxt, text='Amazon',

variable=Sval, value=1, command=next2).place(relx=0.2, rely=0.6)

Radiobutton(nxt, text='Flipkart',

variable=Sval, value=2, command=next3, tristatevalue=0).place(relx=0.2, rely=0.7)

nxt.mainloop()

if \_\_name\_\_ == "\_\_main\_\_":

check\_internet()

headers = {

"User-Agent": 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/96.0.4664.110 Safari/537.36'}

root = Tk()

Sval = StringVar()

urlVal = StringVar()

priceVal = StringVar()

root.geometry("400x250")

root.maxsize(400, 250)

root.minsize(400, 250)

root.title("P.P.T.")

Label(root, text="PRODUCT PRICE TRACKER",

font="comicsansms 13 bold", pady=15).place(relx=0.2, rely=0)

email = Label(root, text="Email")

password = Label(root, text="Password")

email.place(relx=0.2, rely=0.3)

password.place(relx=0.2, rely=0.4)

emailvalue = StringVar()

passwrdvalue = StringVar()

emailentry = Entry(root, textvariable=emailvalue)

passwrdentry = Entry(root, textvariable=passwrdvalue, show='\*')

emailentry.place(relx=0.4, rely=0.3)

passwrdentry.place(relx=0.4, rely=0.4)

Button(text="Submit", background = "gold", command=next1).place(relx=0.6, rely=0.6)

root.mainloop()

**Git Hub Link –**

<https://github.com/Sameer078/ProductPriceTracker>

**TESTING**

**1.**

|  |  |  |  |
| --- | --- | --- | --- |
| Test case ID: TC01 | | Use case ID: UC01 | |
| Test case Title: email and password form generation | |  | |
| Test case description: two text boxes will be displayed where the user needs to enter their email and password | |  | |
| Test steps | Expected result | | Actual result |
| User should click submit button after filling form | Should display the textboxes where the user can enter their email and password | | Textboxes are displayed successfully |

**2.**

|  |  |  |  |
| --- | --- | --- | --- |
| Test case ID: TC02 | | Use case ID: UC02 | |
| Test case Title: website selection | |  | |
| Test case description: user selects a website | |  | |
| Test steps | Expected result | | Actual result |
| Selection of a e-site from the radio-buttons provided | Selected e-site should be displayed | | Selected e-site is displayed |

**3.**

|  |  |  |  |
| --- | --- | --- | --- |
| Test case ID: TC03 | | Use case ID: UC03 | |
| Test case Title: URL and price form generation | |  | |
| Test case description: two text boxes will be displayed where the player needs to enter product URL and desired price | |  | |
| Test steps | Expected result | | Actual result |
| User should click submit button after selection of e-site | Should display the textboxes where the user can enter their URL and desired price | | Textboxes are displayed successfully |

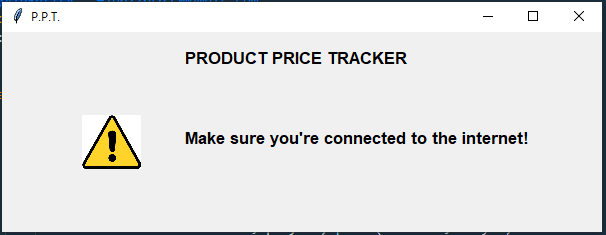
**4.**

|  |  |  |  |
| --- | --- | --- | --- |
| Test case ID: TC04 | | Use case ID: UC04 | |
| Test case Title: Evaluation of the user’s response | |  | |
| Test case description: Message box should be displayed showing mail information | |  | |
| Test steps | Expected result | | Actual result |
| User should click submit button | Should display a message dialog box showing result | | Message box is displayed on whether mail is sent or not |

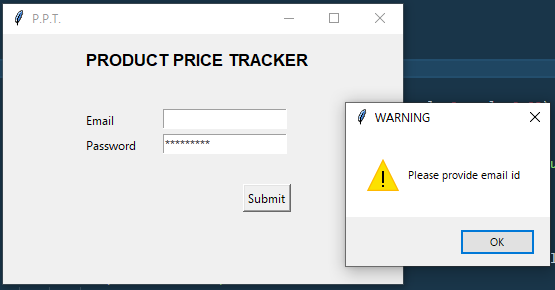
**RESULTS**

**SCREENSHOTS OF APPLICATION TESTCASES**

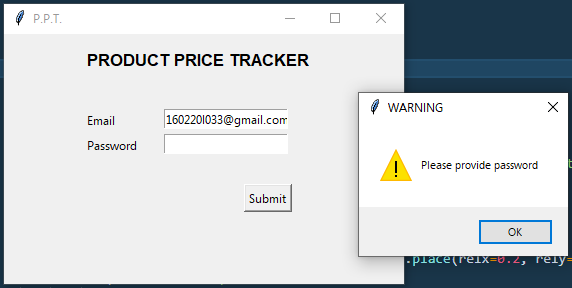
1. If no internet connection



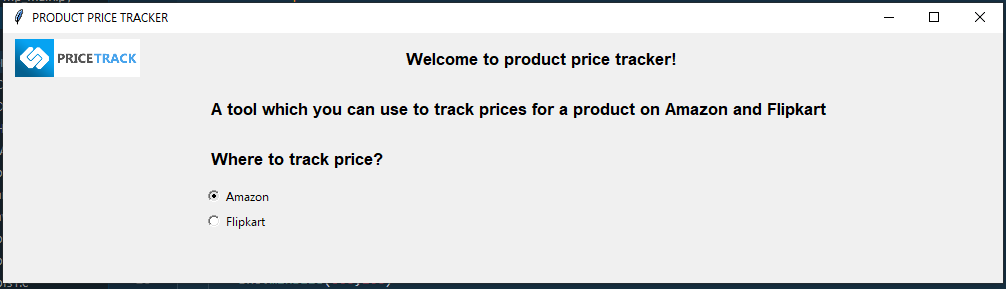
2. With no email

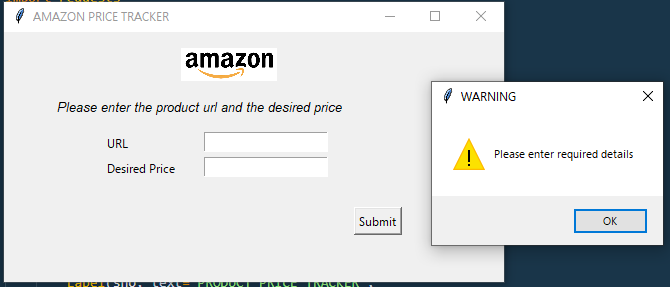


3.With no password

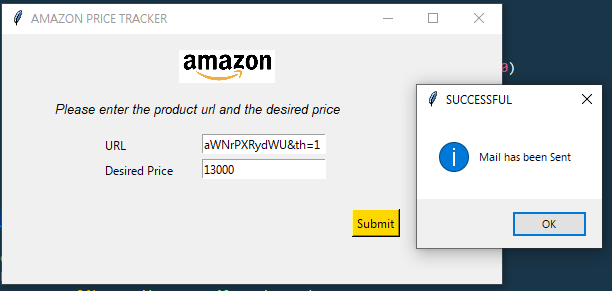


4. Selecting Amazon

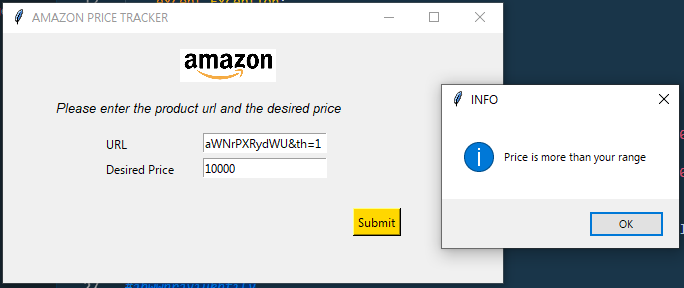
****

4a. Not giving URL or price 

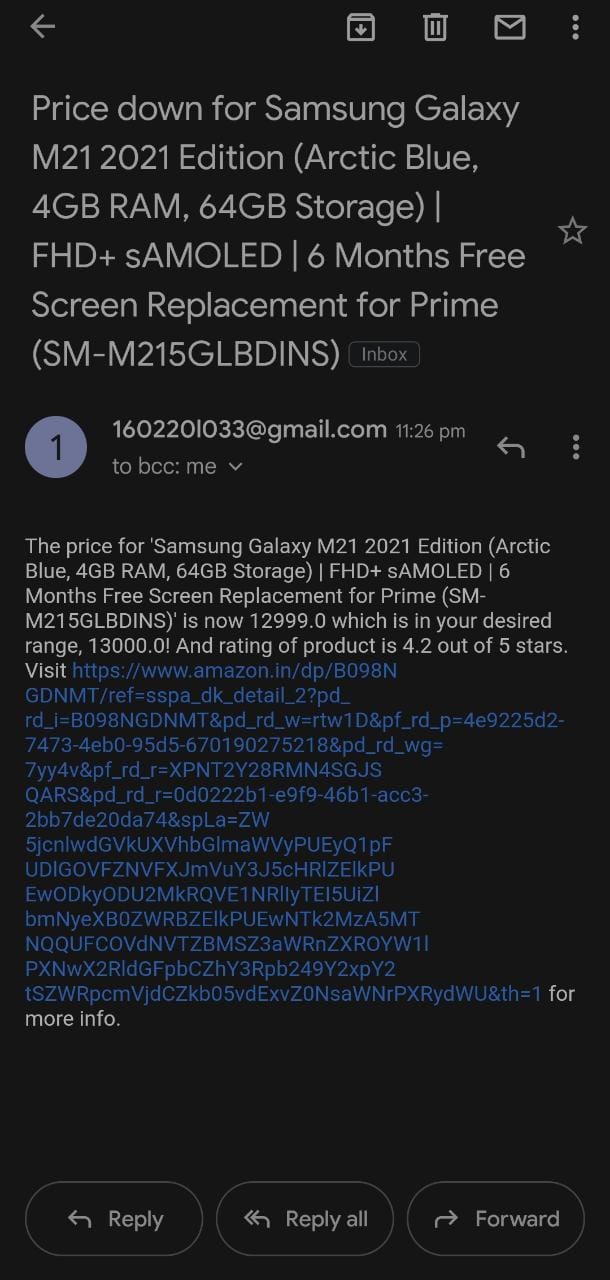
4b. If price < desired price



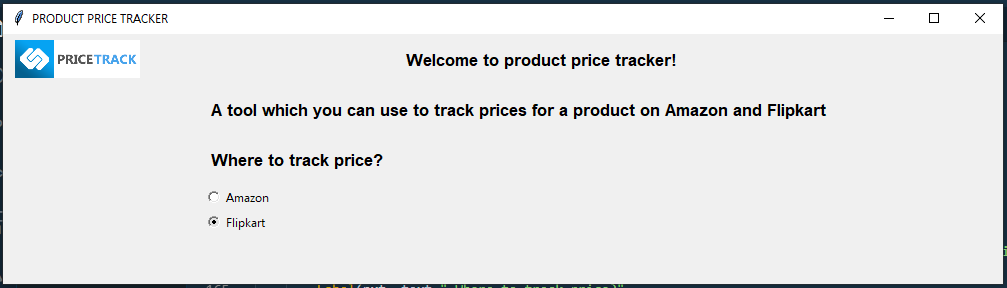
4c. If price > desired price

****

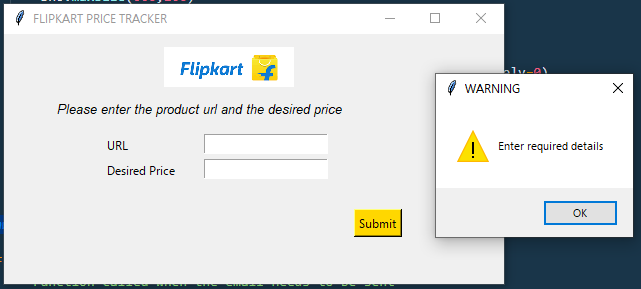
4d. AMAZON Email Demo

****

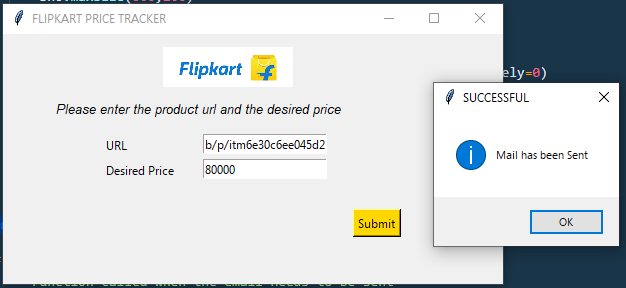
5.Selecting Flipkart

****

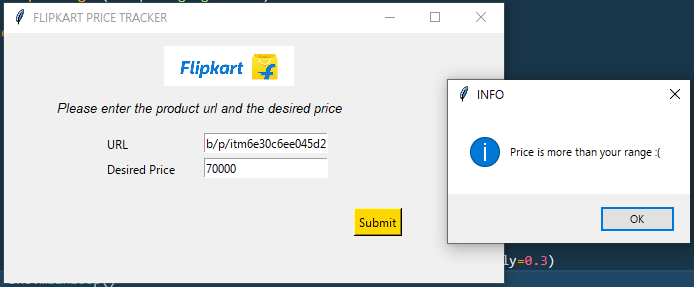
5a. Not giving URL or price

****

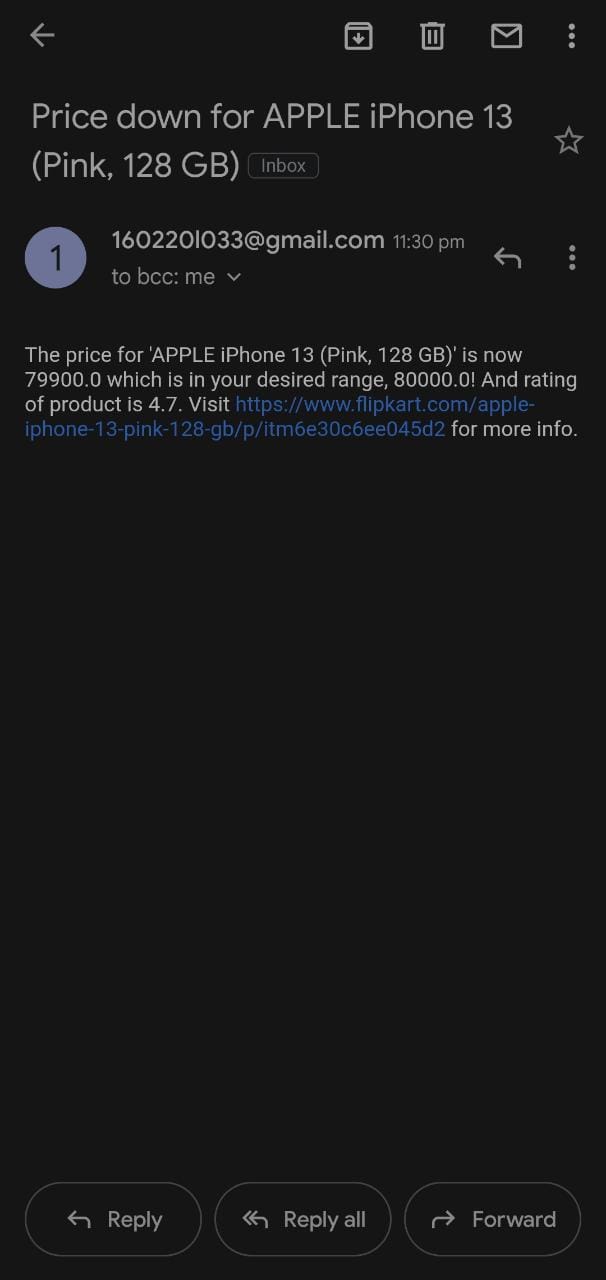
5b. If price < desired price

****

5c. If price > desired price

****

5d. FLIPKART EMAIL demo

****

**ADDITIONAL KNOWLEDGE ACQUIRED**

* Python GUI programming using the module tkinter.
* Using smtplib to send email.
* Use case diagrams.
* Working together as a team
* Using the requests module for getting details from URL.
* Extracting details/data from websites Using beautifulsoup

**CONCLUSION –**

Product Price Tracker is a tool that is used to track the price fluctuations of product and sends

notification in case of a price drop below desired price. This indicates that automated price

tracking solutions are more efficient, feasible, accurate, timely method. The return of investment

by using an automated price tracking tool is quick, simple, and immediately evident. While manual

tracking gets the job done, for the most part, on a very small scale, it is far too limited and is being

outrun by the modern exploits of technology.

Therefore, instead of checking the product details every now and then we can easily track

product data by using this simple tool.

**FUTURE WORK –**

* Add more e-commerce sites
* Creating a database to store its data for every run
* Graphical statistics for better understanding
* Sending alert through sms also (using sinchsms library)
* Add price history chart
* Develop a chrome extension or an App
* Make the whole UI easier to use

**REFERENCES –**

Basic Python

1. Course covered during 2nd semester by K. Srinivasa Chakravarthy Sir.
2. Ppts and handouts provided by the sir.
3. Python Programming - Using Problem Solving Approach First Edition by Reema Thareja.

Tkinter tutorials

1.<https://www.youtube.com/watch?v=YXPyB4XeYLA>

2. <https://youtube.com/playlist?list=PLu0W_9lII9ajLcqRcj4PoEihkukF_OTzA>

3. <https://www.javatpoint.com/python-tkinter>

BeautifulSoup tutorials

1. <https://youtu.be/uufDGjTuq34>
2. <https://www.geeksforgeeks.org/implementing-web-scraping-python-beautiful-soup/>

Smtplib tutorials

1. <https://youtu.be/JRCJ6RtE3xU>
2. <https://www.tutorialspoint.com/python/python_sending_email.htm>

Use cases and activity diagram

1. Hand-outs provided by Mrs C. Sireesha ma’am.
2. <https://www.youtube.com/watch?v=zid-MVo7M-E>
3. <https://www.youtube.com/watch?v=knM8BGY9yVI&t=161s>