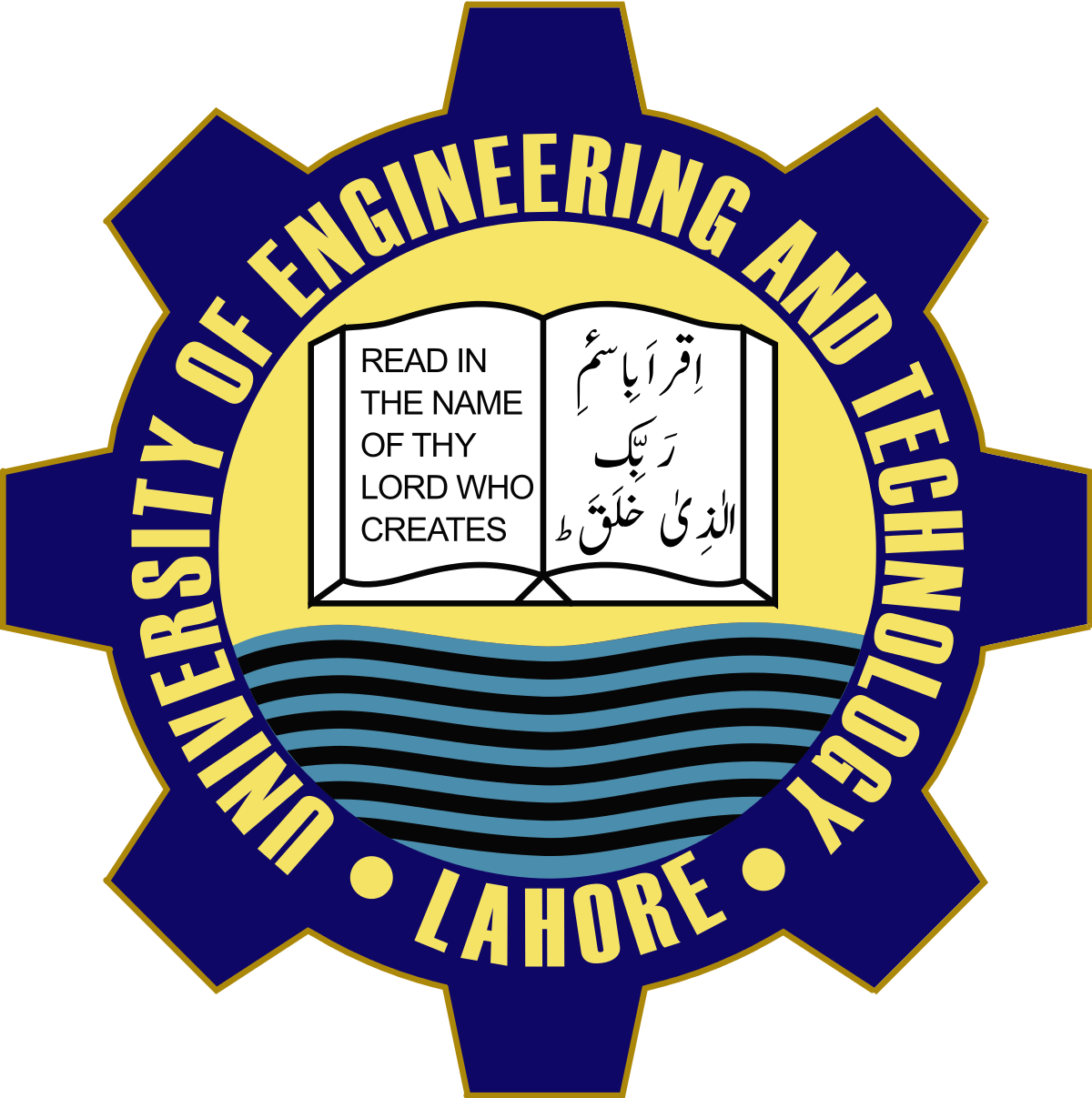
**University of Engineering and Technology**

**Department of Computer Science**

**Course: CS261**

Mid Term Project Report 2021



**Project Supervisor**

Mr. Samyan Qayyum Wahla

| **Submitted By** | **Registration ID** | **Group No.** |
| --- | --- | --- |
| Laibah Azhar | 2020-CS-83 | 65 |

**Github Repository**

CS261F21PID65

**Table Of Contents**

**Pg**

Learning Outcomes----------------------------------------------------------3

Project Description-----------------------------------------------------------3

Resource Management------------------------------------------------------4

* Language and resources used
* Problems Faced

Project Idea-------------------------------------------------------------------5

Data Scraping----------------------------------------------------------------6

* Image of Scraping source
* Code used for scraping

Graphic User Interface------------------------------------------------------8

* Use cases
* Description
* Actors
* Layout in pencil tool
* Actual UI

Sorting Algorithms----------------------------------------------------------14

* Algorithm Description
* Pseudo-code
* Code
* Strengths
* Weaknesses
* Dry Run

Business Case----------------------------------------------------------------31

* Executive Summary
* Target Audience
* Current Process
* Possibilities of this project

**Learning Outcome:**

* Efficiently run Sorting Algorithm
* Use these sorting algorithms on large data set
* Understand Web-Scraping

**Project Description:**

Every second, on average, around 9,685 tweets are tweeted which corresponds to over 580,000 tweets sent per minute and 800 million tweets per day. This makes Twitter an excellent place to get data for projects, as Tweets are an accurate representation of today’s natural language on social media.

Twitter data can be used for research, machine-learning, sales leads, etc. Now a days most news is extracted from twitter. News channels or representatives can use Squeaker to get daily news related to different categories in a single file and assort them with respect to re-tweets.

While working on a project a developer can check the recent trends related to his project or programme and estimate the expected response to his program and check whether he should continue his work on the project or not.

Sports enthusiast can retrieve information of their favorite teams, tournaments or players at one place. Similarly, many games or books enthusiast and a common man would be able to access information and save it in their device with the help of Squeaker.

Total entities of the file to be used would be:

1) Username

2) Handle

3) Time Stamp

4) Comments

5) Likes

6) Re-tweets

7) Text

8) Category

The user can sort information according to two or more options given above, i.e Multi-level searching and multi-column searching would be allowed.

For web scraping **selenium web driver** would be used and the **browser** to be used is **MS-Edge**. To access elements form the web page , **x-path** and its resources would be used.

The files would be saved as .csv files. A separate folder would be made for all of the csv files. The name of the file would be saved by the time stamp, scraping was done at, however it could be changed by the user, according to their preference.

For the UI, pyqt would be used. This program would have its own login interface, to access the app. The user would be required to give their twitter account information to login to twitter, and start with the web scraping.

**Resource Management**

**Project Planning:**

Since I was working alone on this project, I began most of my project work before others, as i had to come up with a project equivalent to those with 2 or more members in their groups.

To accomplish that i started working on it beforehand.

My **routine** before working, was writing down all of the steps i was going to take in order. Followed by actually,doing the work.

The development of the program was in the following steps:

1. Web Scraping on twitter
2. Designing the UI
3. Integrating the UI with a driver code
4. Writing a file including the Sorting Algorithms
5. Integrating all , UI with Sorting Algorithms and driver code.

**Language and Resources used in the project:**

**Python** is used as the primary language, for the web scraping the browser used is **MsEdge,** and **xpath** resources are used for the web scraping. For the ui , **PyQt5 designer is used,**  it was converted to python code by using

pyuic5 filename.ui -o filename.py

The resource file used in the ui were converted to the python code by the following code

pyrcc5 filename.qrc -o filename.py

This project was developed on **Jupyter Notebook.**

**Problems Faced and there Solutions:**

I was faced with many problems, some of them i was able to solve by looking them up on the internet, however there were some i asked about to my friends. For, instance while creating the ui for my project, i could not direct from one ui form to another form. So i asked one of my friends and realised i missed one ‘t’ in QtWidget. Similarly many problems were solved.

However, there were others which i was not able to solve. For example, while scraping twitter, i could not stop the loop, and the scraping keeps going on until the kernel is interrupted or the wifi connection is interrupted.

**The Project Idea:**

While coming up for the project idea, my initial instinct was to scrape products on amazon or any e-commerce websites for an ultimate goal of product hunting. However, my proposal was denied.

Therefore, i looked up for other web scraping ideas and came up with **scraping Twitter,** scarping twitter seemed quite useful, for sales analyst, journalists, or influencers. As in todays age of social media, every thing is posted on the internet and in most cases the first place for updates is twitter and twitter is more reliable compared with other social media platforms like facebook, instagram etc.

Every second, on average, around 9,685 tweets are tweeted which corresponds to over 580,000 tweets sent per minute and 800 million tweets per day. This makes Twitter an excellent place to get data for projects, as Tweets are an accurate representation of today’s natural language on social media.

Twitter data can be used for research, machine-learning, sales leads, etc. Now a days most news is extracted from twitter. News channels or representatives can use Squeaker to get daily news related to different categories in a single file and assort them with respect to re-tweets.

While working on a project a developer can check the recent trends related to his project or programme and estimate the expected response to his program and check whether he should continue his work on the project or not.

Sports enthusiast can retrieve information of their favorite teams, tournaments or players at one place. Similarly, many games or books enthusiast and a common man would be able to access information and save it in their device with the help of Squeaker.

**Data Scraping**

For web scraping selenium web driver would be used and the browser to be used is MS-Edge. To access elements form the web page , x-path and its resources would be used.

**Name of Entities Scraped**

1.Username

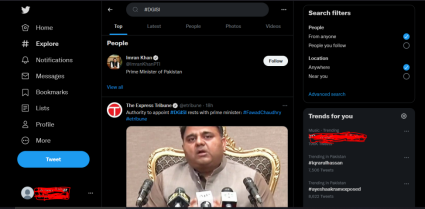
2. Handle

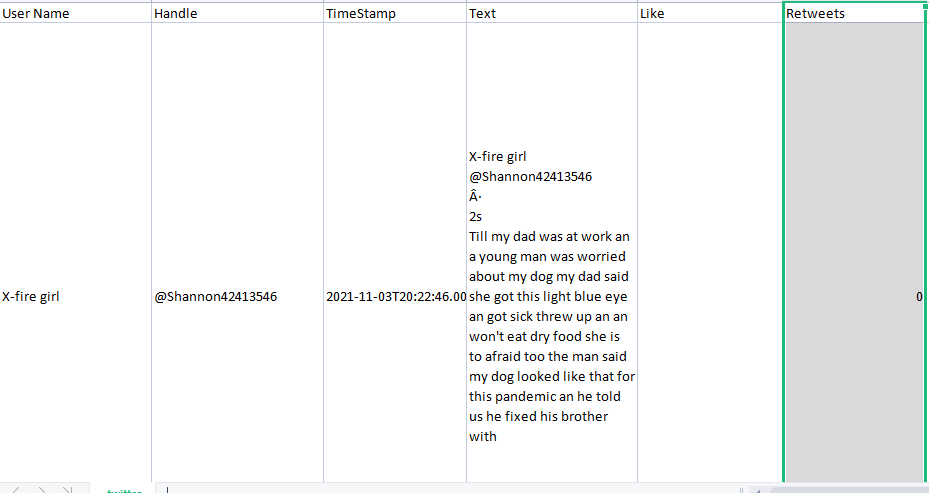
3.Time Stamp

4.Number of Likes

5.Re-tweets

6.Category

**Sample of Scrapping Source** 

**Image of Scraped Data in csv file**

**Code used for scraping:**

*def* start\_scraping():

data = []

tweet\_ids = *set*()

last\_position = driver.execute\_script("return window.pageYOffset;")

scrolling = True

while scrolling:

page\_cards = driver.find\_elements\_by\_xpath('//article[@data-testid="tweet"]')

for card in page\_cards[-1:]:

tweet = get\_tweet\_data(card)

if tweet:

tweet\_id = ''.join(tweet)

if tweet\_id not in tweet\_ids:

tweet\_ids.add(tweet\_id)

data.append(tweet)

scroll\_attempt = 0

while True:

# check scroll position

driver.execute\_script('window.scrollTo(0, document.body.scrollHeight);')

sleep(2)

curr\_position = driver.execute\_script("return window.pageYOffset;")

if last\_position == curr\_position:

scroll\_attempt += 1

# end of scroll region

if scroll\_attempt >= 3:

scrolling = False

save\_in\_csv(data)

driver.close()

break

else:

sleep(2) # attempt another scroll

else:

last\_position = curr\_position

break

print(data)

save\_in\_csv(data)

driver.close()

**Graphic User Interface**

A dark interface has been used to ease the stress on the user's eyes. As most users of this application would be the ones’ who already have a lot of screen time.

| Use Case ID | *U01* |
| --- | --- |
| Name | ***Login*** |
| Actor | *User* |
| Description | *The user can register their account. Once registered they can use their respective username and password to login into their account. While registering the user would be asked to login into their twitter account.* |
| Layout in pencil tool |  |
|  |  |

| Use Case ID | *U02* |
| --- | --- |
| Name | ***Registration*** |
| Actor | *User* |
| Description | *The user can register their account. Once registered they can use their respective username and password to login into their account. While registering the user would be asked to login into their twitter account.* |
| Layout in pencil tool |  |

| Use Case ID | *U03* |
| --- | --- |
| Name | ***Menu*** |
| Actor | *User* |
| Description | *User would be given different options to chose from* |
| Layout in pencil tool |  |

**Sorting Algorithms:**

1.**Time Stamp (**year, month, date, and time**):**

**Algorithm Description:**

The algorithm used would be **Bubble Sort,** for sorting time with respect to year/month/date/time. Bubble sort is an algorithm in which consecutive values/elements of the list are compared with each other, if the first element is greater than the second element they are swapped, otherwise, they remain as is. Then it moves to the next pair.The bubble sort completes, once it passes the whole array without swapping elements.

According to the user’s requirement, the specific year/month/date/time, of the data on Twitter would be scraped, and sorted from oldest to latest. For instance, if the user chooses the year to be 2000, by applying bubble sort the data would be sorted from January to December.

**Pseudo-code of the algorithm:**

1. The timestamps of all data are stored in an array

2. Bubble sort will start with the first element (i=0 index) and check if it is sorted or not,(in case of the month it checks whether their order is correct or not/ for time/date/year it would check whether it is greater than (i+1)th element or not) and sort it.

3. If it is already sorted no swapping takes place

4. Now **i** gets incremented and move to the next element and sorts the next two elements as stated in 3

5. Steps 3 and 4 are repeated till the second last element as the last element is already sorted.

6. Again i is set to 0 and the same steps are repeated to get the 2nd largest element at its respective place.

**Algorithm:**

bubble\_sort(array)

n=length of array

for i=0 to n-1

for j=0 to n-i-1

if array[j]>array[j+1]

swap array[j] with array[j+1]

**Code in Python:**

def bubble\_Sort(a):

n = len(a)

for i in range(n-1):

for j in range(n-i-1):

if a[j]>a[j+1]:

a[j],a[j+1]=a[j+1],a[j]

**Time Complexity Analysis:**

● Comparing each element with the next one for an array of size n takes **n time.**

● The above step is repeated **n times.**

● The best time complexity: O(n^2)

● Average time complexity: O(n^2)

● Worst time complexity: O(n^2)

**Strengths:**

1. Easy to understand

2. Small code

3. Little memory overhead

**Weaknesses:**

1. Takes a lot of time

**Dry Run:**

****



2. **Categories:**

**Algorithm Description:**

The algorithm used would be **Select Sort,** for sorting numbers select sort selects the minimum value and from the un-sorted list and places it at the beginning. In the case of categories, whatever category name is stated by the user, would be selected from the list and brought to the beginning.

The user would enter the category they want, the text of the category would be used and tweets falling in that category would be selected and placed at the beginning

**Pseudo-code of the algorithm:**

1. Find the text same as category text and place it at the beginning

2. Repeat step 1 till all tweets are dealt with.

**Code in Python:**

‘ def selection\_sort(list,category):

n=len(list)

for i in range(n-1):

min\_i=i

for j in range(i+1,n):

if list[j]==Category:

min\_i=j

if min\_i!=i:

list[i],list[min\_i]=list[min\_i],list[i]

**Time Complexity Analysis:**

● There are 2 nested loops, and each is executed **n** times.

● The best time complexity: O(n^2)

● Average time complexity: O(n^2)

● Worst time complexity: O(n^2)

**Strengths:**

1. No overhead memory required

2. As we need specific categories, it will simply select the category name and place in the beginning

3. Performance is influenced by the initial ordering, so we’ll already use twitter’s categories to sort it w.r.t the text, further using sorting algorithm would further refine the search.

**Weaknesses:**

1. Less Efficient

**Dry Run:**

Let C be the category entered by the user

****

3. **Likes:**

**Algorithm Description:**

The algorithm used would be **Insertion Sort.** Virtually the list is split into two sub-arrays and values are taken from the unsorted array and placed in the right order in the sorted sub-array.

In twitter squeaker, the user is provided with an option to sort tweets from maximum to a minimum number of likes or from minimum to a maximum number of likes. So to address both conditions we would just change the code at one position, which would be addressed in the following pseudocode section.

**Pseudo-code of the algorithm:**

1. Begin from the 2nd element and iterate till the last element

2. Compare the present number with the 1st one(or the previous one)

3. If the element in consideration i.e key element is smaller(min-max)/larger(max-min), it is compared with all the previous elements and placed at the correct position.

4. The other elements are pushed ahead to make space for the key element

**Code in Python:**

‘ def insertion\_sort(arr):

n=len(arr)

for i in range(1,n):

key = arr[i]

j=i-1 if (max-min)

while j>=0 and arr[j]>key: while j>=0 and arra[j]<key

arr[j+1]=arr[j]

j=j-1

arr[j+1]=key

**Time Complexity Analysis:**

● If the list is already sorted in the desired order, only the first loop runs n-1 times

● The best time complexity: O(n)

● There are 2 nested loops, and each is executed **n** times.

● Average time complexity: O(n^2)

● Worst time complexity: O(n^2)

**Strengths:**

1. Simple Code

2. Minimum Space required

**Weaknesses:**

1. Does not perform as good as other algorithms

2. Does not work well with huge amount of lists

**Dry Run:**

For min-max number of likes

****

4. **Re-tweets:**

**Algorithm Description:**

The algorithm used would be **Quick Sort.** It is a divide and conquer algorithm. It takes an element as a pivot, in our algorithm we’ll use the last element as a pivot. We use a function partition(), and let’s say our pivot is x, we’ll place x at its correct position and with smaller elements behind and greater elements ahead of x.

In accordance with the user’s requirement, we’ll sort tweets in either order of max-min or min-max re-tweets.

**Pseudo-code of the algorithm:**

1. Using partition function we split the array into two halves with our pivot which is the last element of the list at its correct position

2. Recursively we’ll apply the quick sort function on the split arrays the same way as done before.

**Code in Python:**

‘ def quickSort(arr,low,high):

if low<high:

pi= partition(arr,low,high)

quickSort(arr,low,pi-1)

quickSort(arr,pi+1,high)

def partition(arr,low,high):

pivot=arr[high]

i=low-1

for j in range(low,high):

if arr[j]<=pivot:

i=i+1

arr[i],arr[j]=arr[j],arr[i]

arr[i+1],arr[high]=arr[high],arr[i+1]

return i+1

**Time Complexity Analysis:**

● Partitioning elements takes n time

● The problem is divided by factor 2

● The best time complexity: O(nlogn)

● Average time complexity: O(nlogn)

● When an array is sorted it will give us the worst time complexity

● Worst time complexity: O(n^2)

**Strengths:**

1. Can easily deal with a huge list

2. No overhead storage is required

**Weaknesses:**

1. If the list is already sorted it gives the worst case

**Dry Run:**

For min-max number of retweets.



5. **Multi-level Sorting:**

**Algorithm Description:**

The algorithm used would be **Merge Sort.** It is also a divide and conquer algorithm. It divides the array into sub-arrays in every iteration into two equal sub-arrays and sorts them by calling itself recursively. In the end, it merges the sorted halves.

**Pseudo-code of the algorithm:**

1. **Divide:** Finds the midpoint position and divides the aray

2. **Conquer:** Recursively it sorts the sub-arrays, if q is the midpoint then it recursively calls itself with array(....q) and (q+1…)

3. **Combine:** The sorted halves are combined

**Code in Python:**

‘ def merge\_sort(list):

if len(list)>1:

mid=len(list)//2

left = list[:mid]

right=list[mid:]

merge\_sort(left)

merge\_sort(right)

i=0

j=0

k=0

while i<len(left) and j<len(right):

if left[i]<=right[j]:

list[k]=left[i]

i+=1

else:

list[k]=right[j]

j+=1

k+=1

while i<len(left):

list[k]=left[i]

i+=1

k+=1

while j<len(right):

list[k]=right[j]

j+=1

k+=1

**Time Complexity Analysis:**

● The best time complexity: O(nlogn)

● Average time complexity: O(nlogn)

● Worst time complexity: O(nlogn)

**Strengths:**

1. Can sort of files of any size

**Weaknesses:**

1. Requires extra space

**Dry Run:**

**** 

**Business Case:**

**Executive Summary:**

Twitter is a social media platform, founded in March 21, 2006, by [Jack Dorsey](https://www.google.com/search?client=firefox-b-d&sxsrf=AOaemvIM4-1qfJ_dufmg2ALU9BgXzdGkFQ:1636130634581&q=Jack+Dorsey&stick=H4sIAAAAAAAAAOPgE-LSz9U3yMgzLEtOVwKzTSotkvLMtNSzk630k0qLM_NSi4vhjPj8gtSixJLM_DyrtPzSvJTUokWs3F6JydkKLvlFxamVO1gZAetVgUJUAAAA&sa=X&ved=2ahUKEwj3u57K1YH0AhUCKBoKHUEtCkkQmxMoAXoECE0QAw), [Evan Williams](https://www.google.com/search?client=firefox-b-d&sxsrf=AOaemvIM4-1qfJ_dufmg2ALU9BgXzdGkFQ:1636130634581&q=Evan+Williams&stick=H4sIAAAAAAAAAOPgE-LSz9U3yMgzLEtOV-IEsc2STHOStdSzk630k0qLM_NSi4vhjPj8gtSixJLM_DyrtPzSvJTUokWsvK5liXkK4Zk5OZmJucU7WBkBlWwfbFUAAAA&sa=X&ved=2ahUKEwj3u57K1YH0AhUCKBoKHUEtCkkQmxMoAnoECE0QBA), [Noah Glass](https://www.google.com/search?client=firefox-b-d&sxsrf=AOaemvIM4-1qfJ_dufmg2ALU9BgXzdGkFQ:1636130634581&q=Noah+Glass&stick=H4sIAAAAAAAAAOPgE-LSz9U3yMgzLEtOV-LWT9c3NDKoircwM9VSz0620k8qLc7MSy0uhjPi8wtSixJLMvPzrNLyS_NSUosWsXL55SdmKLjnJBYX72BlBABmFgofVAAAAA&sa=X&ved=2ahUKEwj3u57K1YH0AhUCKBoKHUEtCkkQmxMoA3oECE0QBQ), [Biz Stone](https://www.google.com/search?client=firefox-b-d&sxsrf=AOaemvIM4-1qfJ_dufmg2ALU9BgXzdGkFQ:1636130634581&q=Biz+Stone&stick=H4sIAAAAAAAAAOPgE-LSz9U3yMgzLEtOVwKzTcwLSgottNSzk630k0qLM_NSi4vhjPj8gtSixJLM_DyrtPzSvJTUokWsnE6ZVQrBJfl5qTtYGQE7MdPPUgAAAA&sa=X&ved=2ahUKEwj3u57K1YH0AhUCKBoKHUEtCkkQmxMoBHoECE0QBg). It is a huge platform, used by billions of people around the world. As of today i.e 11/5/21 around 9,723 tweets per second that means 583,380 tweets per minute, and 35,002,800 tweets per hour, giving us 840,067,200 tweets per day.

In today’s world where all kinds of news from around the world is accessible just at the tip of our hands through these social media platforms, twitter comes at the top. It is considered faster and way better than other social media platforms.

Having this information in an organized form on your mobile phones can be revolutionary.

**Target Audience:**

Our target audience would be **Marketing** and **Sales Analyst, influencers, Journalists.**

**Marketing and Sales Analyst:**

They have to understand the market trends related to their present and upcoming projects. To graph out their future income and profits, they need to understand what are the present and possible trends amongst their clients. What better way is there than to check through social media where the consumers pour out their hearts, and express their likes and dislikes..

Nowadays a common man is quick to review whatever new product they try and post it on twitter, therefore twitter can be useful tool for them. However, solely using twitter filtering out the useful information could be tiresome, to solve that problem the solution would be use **twitter squeaker,**  and scarpe the important data out.

**Influencers and Journalists:**

Influencers influence the audience in many ways, but for them to stay influential they need to be aware of the present hashtags, and keep their followers updated with daily buzz. That’s where having a twitter scraping program can help them out.

Likewise, journalists also require hot news, related to their genre of journalism. They too can get recent and trend setting information from twitter. To make it easier for them **twitter squeaker** could be a useful tool.

**Current Process:**

Currently, the ui of the project has been decided on, basic sorting has been implemented and integrated with the program. Scraping of twitter is also achieved.

**Possibilities of this Project:**

Fixing the basic problems of the project would be enough for the twitter squeaker to progress. However by adding some additional and useful upgrades, and enhancing the outreach of this project. These upgrades may include:

1. Alert: set by the user, e.g “If a tweet about cricket has more than 100 likes and 100 retweets” alert the user.
2. The information is given as a preview first and it would only be saved if the user enters the save button
3. A secure folder for all the saved files of the user, so that they are only accessible by the user.