**AUTOMATIC TOPIC TAGGING FOR DOCUMENTS USING AN API**

1. Introduction

a. Overview

b. Purpose

2. Literature Survey

a. Existing problem

b. Proposed solution

3. Theoretical analysis

a. Block diagram

b. Hardware/Software designing

4. Experimental Investigations

5. Flowchart

6. Result

7. Advantages &amp; Disadvantages

8. Applications

9. Conclusion

10. Future Scope

11. Bibliography

12. Appendix

a. Source code

b. UI output screenshots

**1. Introduction:**

a. Overview:

Topic modeling is a machine learning technique that automatically analyzes text data to determine cluster words for a set of documents. This is known as ‘unsupervised’ machine learning because it doesn’t require a predefined list of tags or training data that’s been previously classified by humans.

b. Purpose:

Businesses deal with large volumes of unstructured text every day. Think about all of the customer interactions and brand mentions in emails, support tickets, social media posts, online feedback and reviews, and other information that an organization sends and receives. The list is endless. When it comes to analyzing huge amounts of text data, it’s just too big a task to do manually. It’s also tedious, time-consuming, and expensive. Manually sorting through large amounts of data is more likely to lead to mistakes and inconsistencies. Plus, it doesn’t scale well.

**2. Literature Survey:**

a. Existing problem:

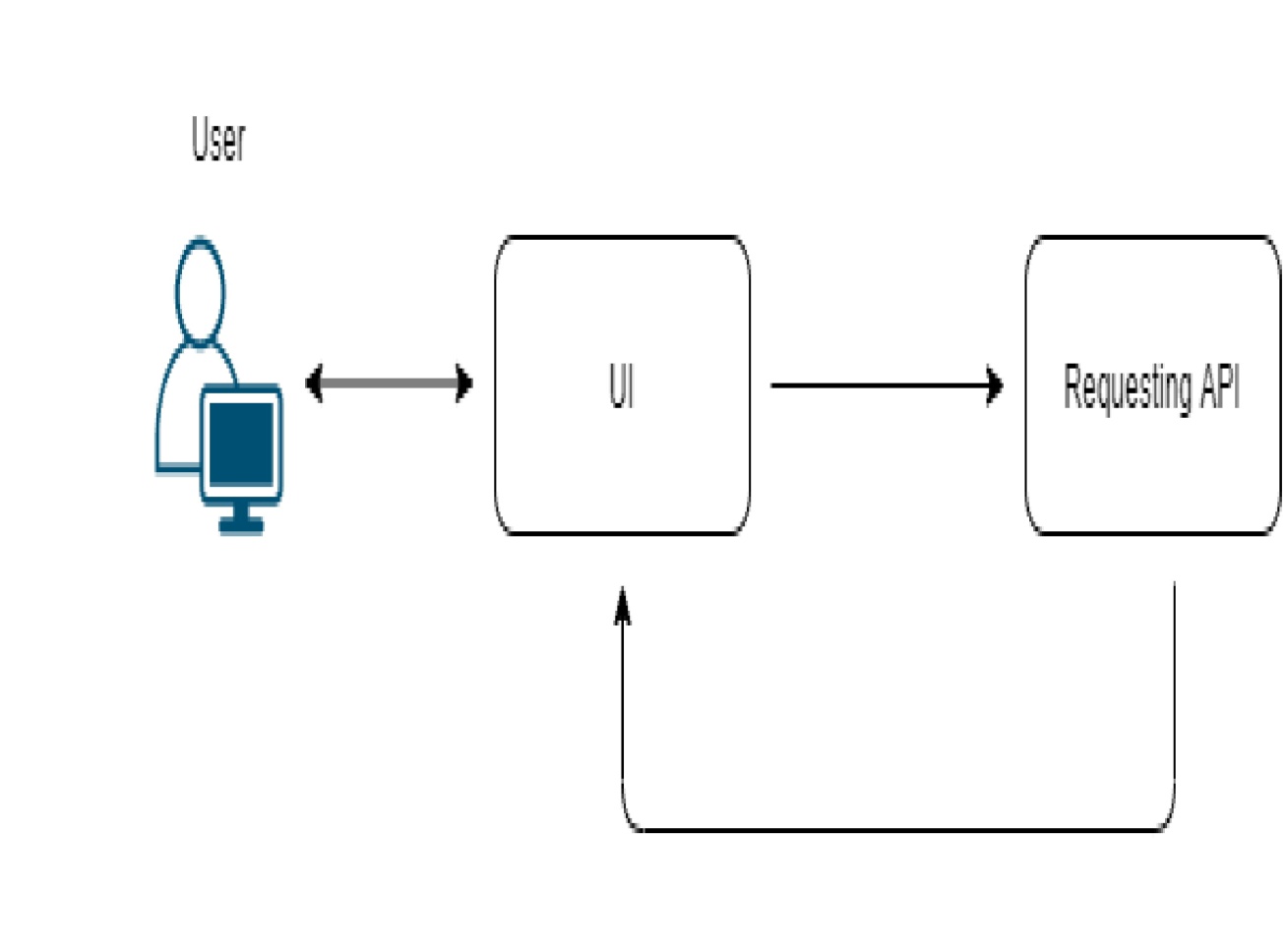
if you work in a company that processes hundreds, or even thousands of customer interactions every day. Data Analysis of social media posts, emails, chats, open-ended survey responses, and more, is not an easy task, and less so when delegated to humans alone.

b. Proposed solution:

Topic analysis is a Natural Language Processing (NLP) technique that allows us to automatically extract meaning from texts by identifying recurrent themes or topics.Topic analysis models enable you to sift through large sets of data and identify the most common and most important topics in an easy, fast, and completely scalable way.

**3. Theoretical Analysis:**

a. Block diagram:



b. Software Designing:

To complete the project successfully, you need to install following software & packages:

**Activity 1: Install Anaconda IDE / Anaconda Navigator.**

* In order develop a solution to this problem statement, we need an environment to write and test the code.
* We use Anaconda IDE (Integrated Developing Environment).
* The below link helps to download & install Anaconda Navigator.

**Link:** [**https://www.youtube.com/watch?v=5mDYijMfSzs**](https://www.youtube.com/watch?v=5mDYijMfSzs)

**Activity 2: Installation of Python Packages**

* To write a code for video streaming, we need to install Open CV (open computer vision) library. Follow the below steps to install it.

* Open Anaconda Navigator as administrator.
* Type “pip install requests” and press enter.

**4. Experimental Investigations:**

we have completed the project by following the below steps:

**Activity 1: Rapid API Account Creation.**

Now, the activities 1&2 is all about creation of an account in the Rapid API platform, to generate desired application oriented API.

Browse the below link to navigate to Rapid API platform.

Link: <https://rapidapi.com/>

Go through the below link for the complete process.

Link: <https://www.youtube.com/watch?v=RcSCo5unPrQ>

**Activity 2: Subscription of Application Oriented API**

In this activity, you need to subscribe the desired application.

* Navigate to the below link, to subscribe for the Topic Tagging API.

**Link:** <https://rapidapi.com/twinword/api/topic-tagging>

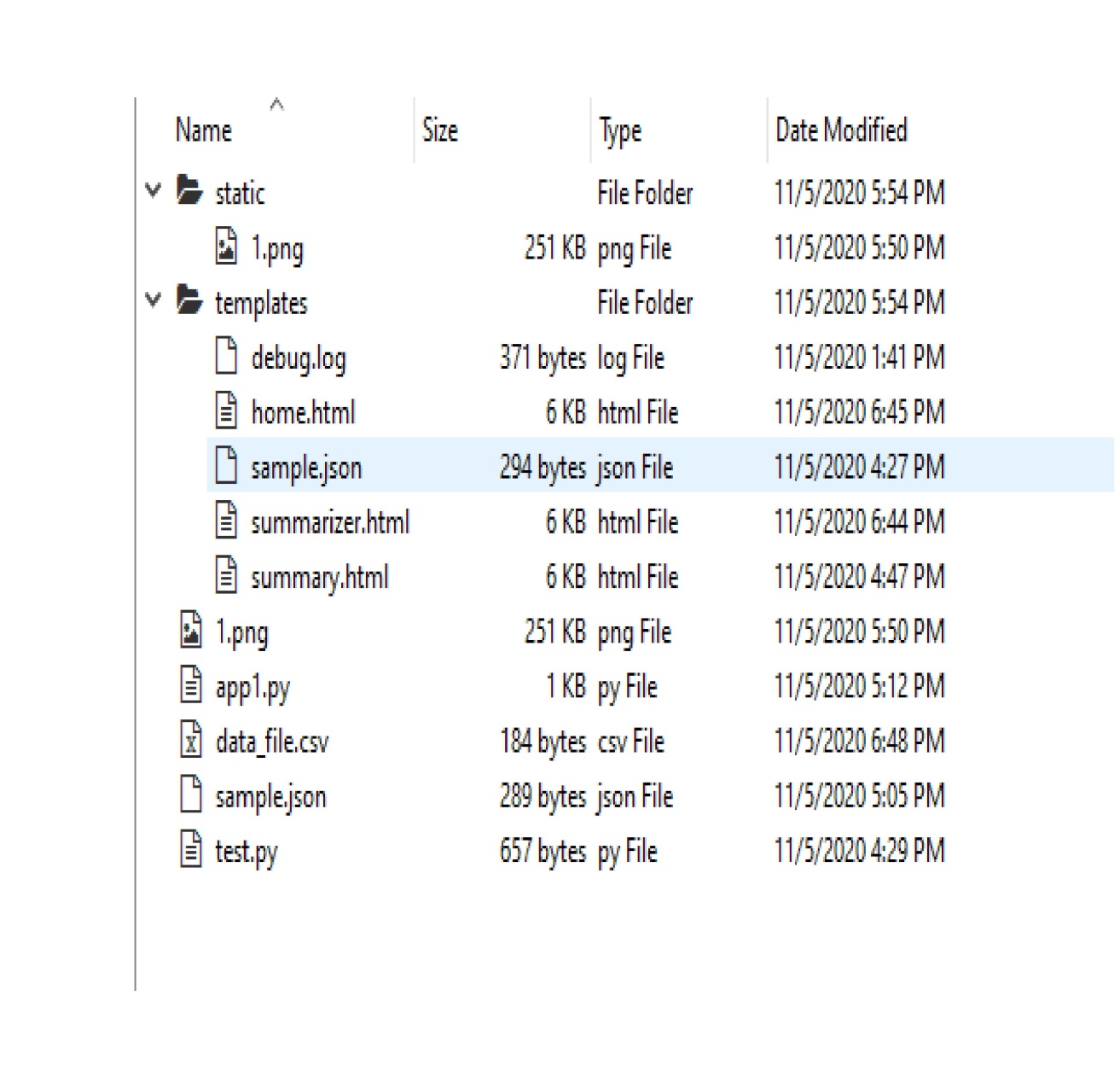
* Copy the python code and API details to integrate to Flask Application.

**Note:** Go through the reference link provided in the activity 2, to subscribe for the desired API.

**Activity 3: Building a Flask Application**

In this activity, you was developing an html page and integrate the generated API.

* In order to process proceed with this activity, arrange all your project files in the below format.



* All the above files are used to develop a flask application.
* In the static folder we will be storing all the flask background images.
* In the templates, you will storing all the rendering files and html pages.
* In the app1.py files, you will be writing the python script for the flask application to integrate to user interface.
* In the test.py , you will be writing a code to save csv & json files.
* You will storing all the probabilities of the output values in the form of csv (comma separated value i.e. data\_file.csv) & will be stored in the form of json.

**5. Project Flow/Flow chart:**

1. Installation of Pre-requisites

->Installation of Anaconda IDE / Anaconda Navigator.

->Installation of Python packages.

2. Rapid API Account Creation.

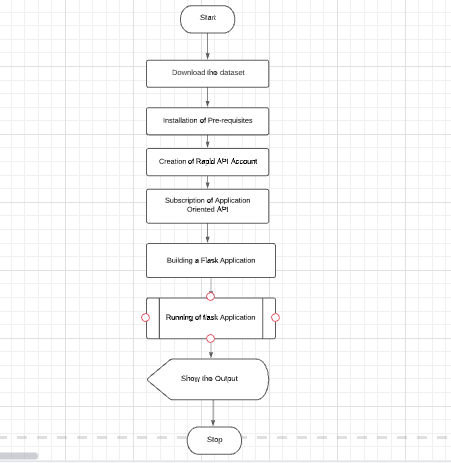
3. Subscription of Application Oriented API.

4. Building a Flask Application.

->Importing of Libraries and routing the html pages

->Running of flask Application.

Flow chart:



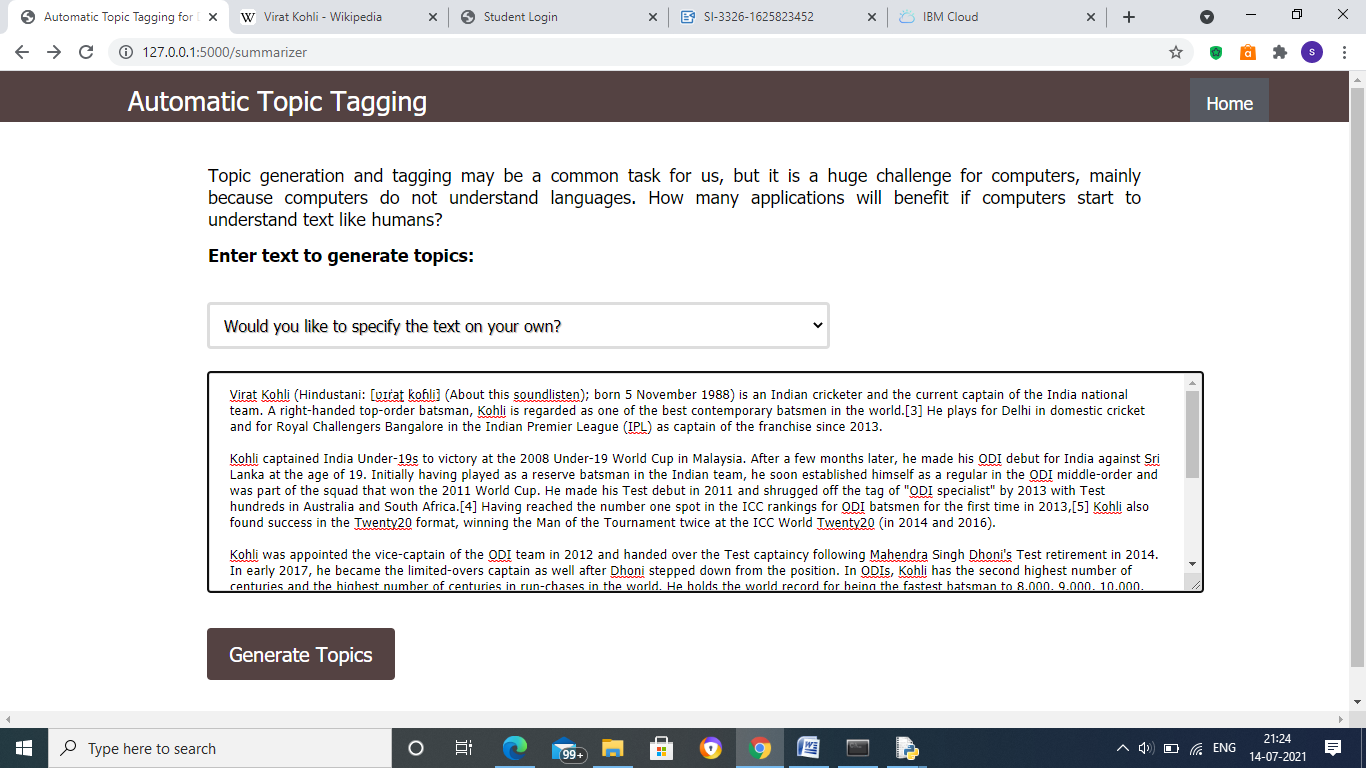
**6. Result:**

* We have successfully completed creating of automatic topic tagging application.
* Our application about detecting the topics and their respective probabilities.
* We have give the some text as input to our application then it is detects the topics as well as their probabilities respectively.

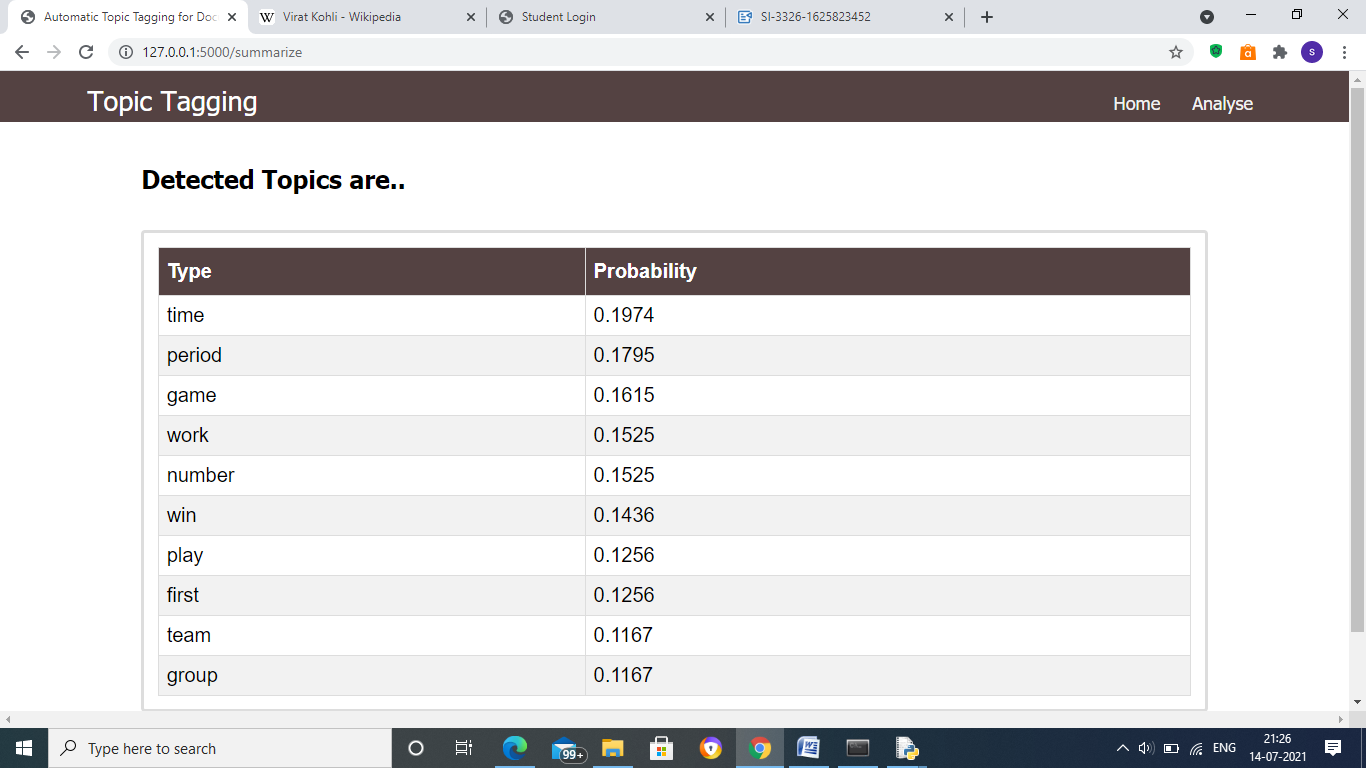
Our application app name is app1.it is a python file.

**The below screenshots shows the outputs of topic tagging:**

1.The below screenshot shows the input taking of our application



The below picture shows the output :



**7. Advantages &Disadvantages:**

Advantages of topic tagging**:**

**** Browsability

Browsing is extremely fruitful using social tags and folksonomies. As mentionedMathes (2004), that tags allow us to see what resources others have found useful

what things these resources might be useful for, and what other terms you might want to search under. This could be seen from Delicious site that is helpful if a user accesses information freely. Through others' personal pages and the "popular" page, users can get a sense of what other people find interesting. By browsing specific people and tags, users can find web resources

that are of interest to them and can find people who have common interests.

**** Tagging reflects users' vocabulary

Tagging directly represents the vocabulary of users, because they are the ones creating the metadata. Popular tags show us what terms are preferred by the group. This can lead to the creation of new controlled vocabularies that might be easier to search because they would better reflect users' language. Kroski said (2005) put another way, folksonomies are inclusive because users are participants and experts, and because users see themselves reflected in the metadata.

**** Classification of large amounts of content at little cost

Social tagging takes advantage of the large number of users participating.

Because the more users who tag, the more data can be tagged. If we compare to traditional metadata creation by information professionals, it would not be able to come close to the productivity of taggers, especially for the huge

amount of information that is on the Web. Tagging distributes the

responsibility and increases productivity.

**** Current and flexible

All information that are displayed on the Delicious site as a social bookmarking site indicate a real time information and flexibility. This as mentioned by Kroski (2005) that folksonomies are fluid by their very nature. Users can tag and untag data whenever they want with speed and ease, meaning that folksonomies remain current and reflect current viewpoints.

**** Community

Interaction of the people in accessing social tagging site indicates virtual

community. Social tagging can bring people together. Not only are all taggers bound together by the shared goal of classifying information, but taggers can see who else shares their interests and their vocabulary (Kroski 2005). This community also will identify users' interests. Users' lists of tags can be considered descriptive of the interests they hold as well as of their method of classifying those interests. Users' tag lists grow over time, as they discover new interests and add new tags to categorize and describe them. It is possible that the newly growing tag represents a new interest or category to the user (Golder and Huberman 2005).

Disadvantages of tagging **:**

**** Lack of control

Social tagging always allows people to tag freely on the site. User ability is

different in writing a vocabulary on the tag. This disadvantage is the issue that led to the use of controlled vocabularies for subject cataloguing in libraries. Kroski said (2005) that social tagging lets users use whatever tags they want, meaning that users may tag the same item any number of different ways.,

**** Lack of recall

A lack of control affects to a lack of recall. When we search of tags using the

search term "dog" will not pick up items tagged with synonyms, which means that the search is only finding a small fraction of the information available on the subject. This also mentioned by Kroski (2005) that the lack of vocabulary control leads to a lack of recall.

**** Lack of precision

A lack of control and recall affect to a lack of precision. Users may not have

tagged items with relevant tags, or tagged too broadly, or tagged in such a way that is too personal, leaving the user with irrelevant material.

**8. Applications:**

Now a days Automatic Topic tagging application used in many sites,apps etc.some of the application are:

* Electronic mails
* Online feedback and reviews
* Social media posts
* Facebook
* Instagram
* Twitter
* Wikipedia
* Google
* Mobile apps
* Many sites etc.

**9. Conclusion:**

topic tagging not only change the methodology of classification in term of retrieval information, but also change in the way that classifiers organize information. It has removed all concept of hierarchy from the scheme of knowledge organization, facilitating knowledge discovery and web indexing.

Although, topic tagging is not a perfect system in the world, but it has help users in browsing information and classify a lot of contents at little cost, as well as creating a new virtual community that can identify user`s interest. A number of disadvantages of social bookmarking can be a lesson to all information expert to create the best formula in the future. As a result, by controlling vocabularies, search engines could present search results in clusters and attach each cluster to terms having the highest frequency, designating them as the tagging terms of the luster. In addition these advantages allowing users to tell a system what method of finding digital content works for them is a good start to having a truly

interactive and responsive information retrieval system.

**10. Future Scope:**

Businesses deal with large volumes of unstructured text every day. Think about all of the customer interactions and brand mentions in emails, support tickets, social media posts, online feedback and reviews, and other information that an organization sends and receives. The list is endless. When it comes to analyzing huge amounts of text data, it’s just too big a task to do manually. It’s also tedious, time-consuming, and expensive. Manually sorting through large amounts of data is more likely to lead to mistakes and inconsistencies. Plus, it doesn’t scale well.

Day by day the data in the cloud or personal storage incresing a lot. so always topic tagging places huge role for reading those data.

**11. Bibliography:**

To complete the project successfully, you need to install following software & packages:

**Activity 1: Install Anaconda IDE / Anaconda Navigator.**

* In order develop a solution to this problem statement, we need an environment to write and test the code.
* We use Anaconda IDE (Integrated Developing Environment).
* the below link used to download & install Anaconda Navigator.

**Link:** [**https://www.youtube.com/watch?v=5mDYijMfSzs**](https://www.youtube.com/watch?v=5mDYijMfSzs)

**Activity 2: Installation of Python Packages**

* To write a code for video streaming, we need to install Open CV (open computer vision) library. Follow the below steps to install it.

* Open Anaconda Navigator as administrator.
* Type “pip install requests” and press enter.

The above steps allows to install the opencv packages.

**Activity 3: Rapid API Account Creation.**

Now, the activities 3&4 is all about creation of an account in the Rapid API platform, to generate desired application oriented API.

Browse the below link to navigate to Rapid API platform.

Link: <https://rapidapi.com/>

Go through the below link for the complete process.

Link: <https://www.youtube.com/watch?v=RcSCo5unPrQ>

**Activity 4: Subscription of Application Oriented API**

In this activity, you need to subscribe the desired application.

* Navigate to the below link, to subscribe for the Topic Tagging API.

**Link:**<https://rapidapi.com/twinword/api/topic-tagging>

* Copy the python code and API details to integrate to Flask Application.

**Note:** Go through the reference link provided in the activity 4, to subscribe for the desired API.

**Activity 5: Building a Flask Application**

In this activity, you will be developing an html page and integrate the generated API.

The below Github link all the files of this project available:

https://github.com/smartinternz02/SI-GuidedProject-3326-1624018019

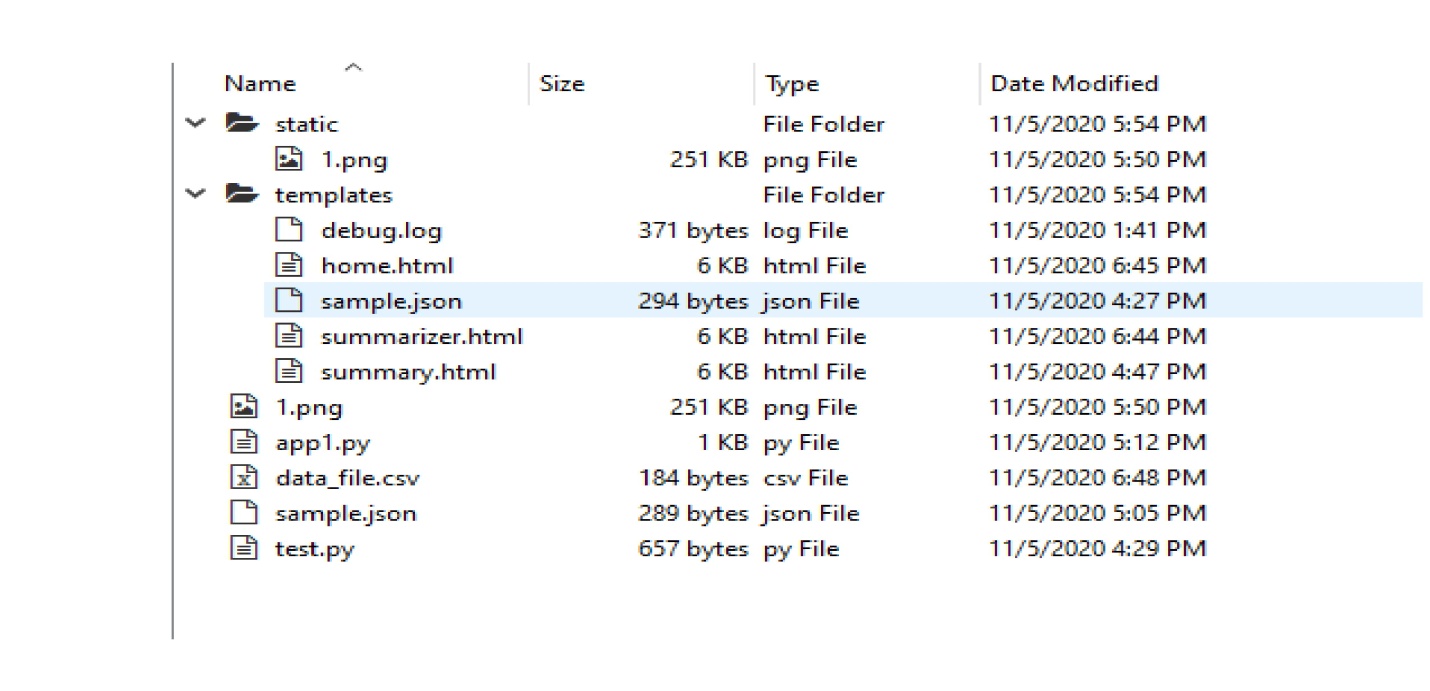
**12. Appendix**

**a. Source code:**

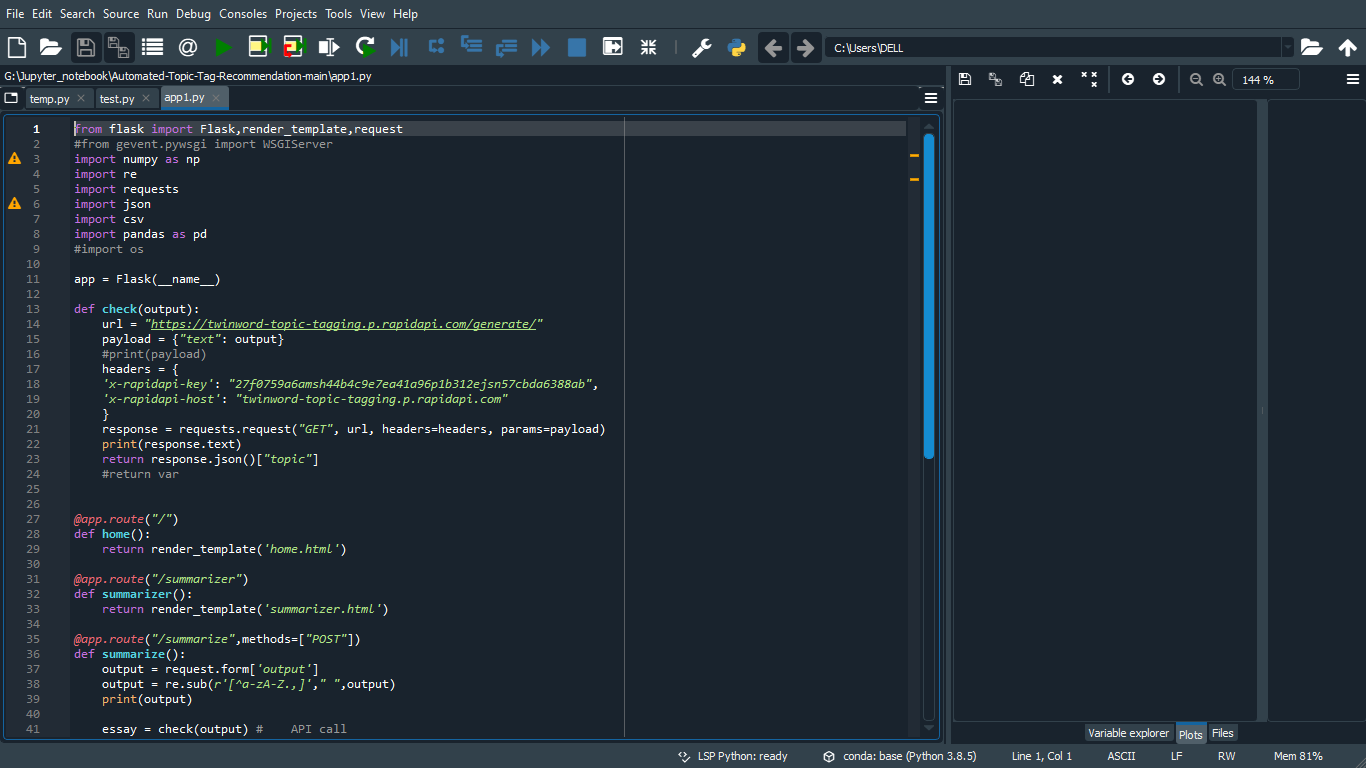
* In the static folder we will be storing all the flask background images.
* In the templates, you will storing all the rendering files and html pages.
* In the app1.py files, you will be writing the python script for the flask application to integrate to user interface.
* In the test.py , you will be writing a code to save csv & json files.

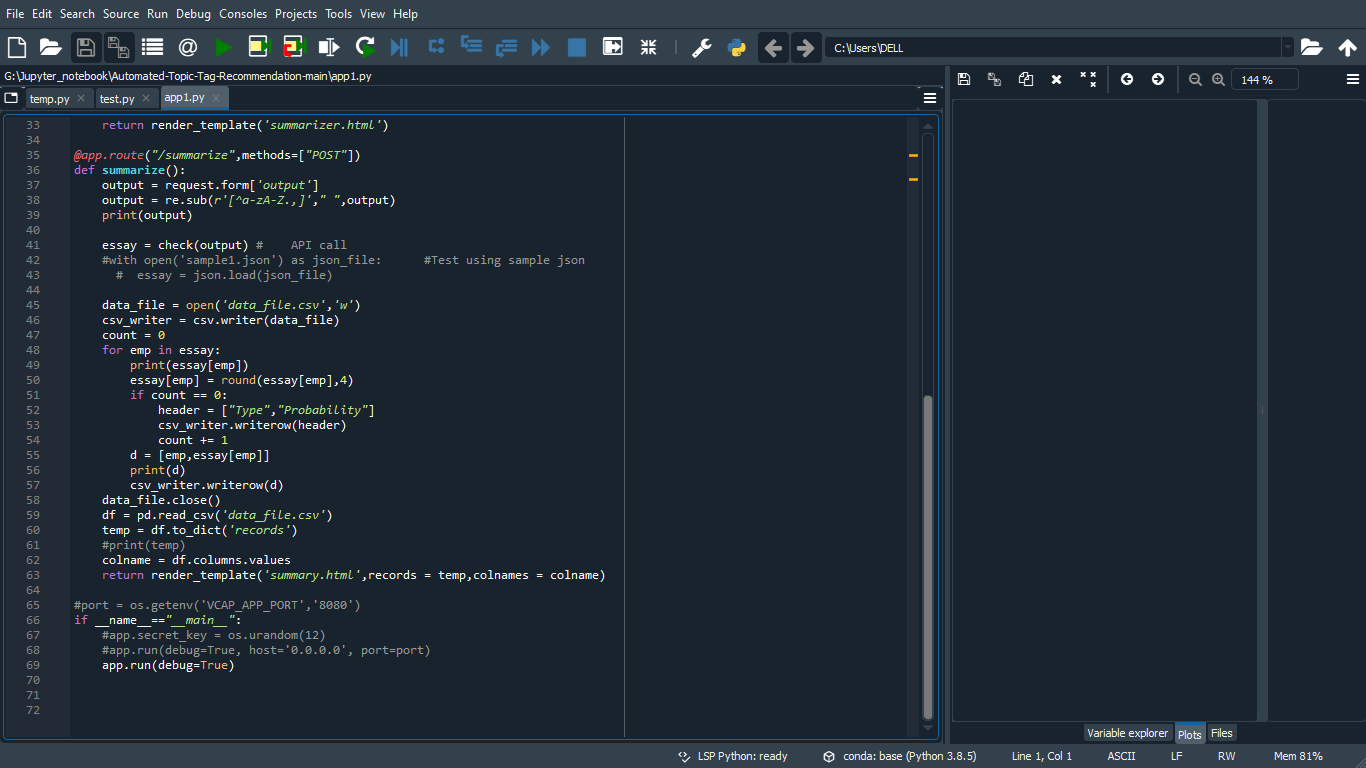
You was storing all the probabilities of the output values in the form of csv (comma separated value i.e. data\_file.csv) & will be stored in the form of json

* In order to process proceed with this milestone, arrange all your project files in the below format.



The below one shows the code of app1.py:





code:

from flask import Flask, render\_template, request

#from gevent.pywsgi import WSGIServer

import numpy as np

import re

import requests

import json

import csv

import pandas as pd

#import os

app = Flask(\_\_name\_\_)

def check(output):

url = "https://twinword-topic-tagging.p.rapidapi.com/generate/"

payload = {"text": output}

#print(payload)

headers = {

'x-rapidapi-key': "27f0759a6amsh44b4c9e7ea41a96p1b312ejsn57cbda6388ab",

'x-rapidapi-host': "twinword-topic-tagging.p.rapidapi.com"

}

response = requests.request("GET", url, headers=headers, params=payload)

print(response.text)

return response.json()["topic"]

#return var

@app.route("/")

def home():

return render\_template('home.html')

@app.route("/summarizer")

def summarizer():

return render\_template('summarizer.html')

@app.route("/summarize",methods=["POST"])

def summarize():

output = request.form['output']

output = re.sub(r'[^a-zA-Z.,]'," ",output)

print(output)

essay = check(output) # API call

#with open('sample1.json') as json\_file: #Test using sample json

# essay = json.load(json\_file)

data\_file = open('data\_file.csv','w')

csv\_writer = csv.writer(data\_file)

count = 0

for emp in essay:

print(essay[emp])

essay[emp] = round(essay[emp],4)

if count == 0:

header = ["Type","Probability"]

csv\_writer.writerow(header)

count += 1

d = [emp,essay[emp]]

print(d)

csv\_writer.writerow(d)

data\_file.close()

df = pd.read\_csv('data\_file.csv')

temp = df.to\_dict('records')

#print(temp)

colname = df.columns.values

return render\_template('summary.html',records = temp,colnames = colname)

#port = os.getenv('VCAP\_APP\_PORT','8080')

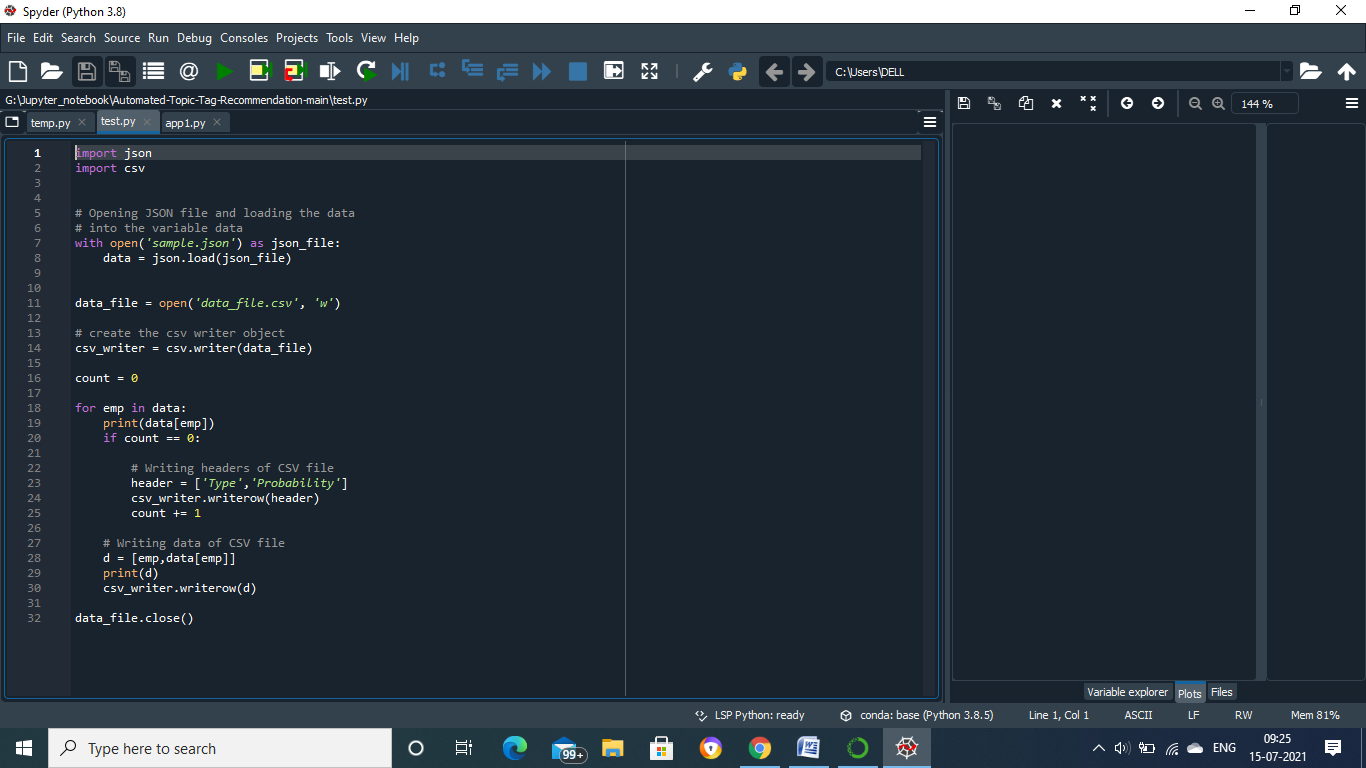
if \_\_name\_\_=="\_\_main\_\_":

#app.secret\_key = os.urandom(12)

#app.run(debug=True, host='0.0.0.0', port=port)

app.run(debug=True)

test.py:



code:

import json

import csv

# Opening JSON file and loading the data

# into the variable data

with open('sample.json') as json\_file:

data = json.load(json\_file)

data\_file = open('data\_file.csv', 'w')

# create the csv writer object

csv\_writer = csv.writer(data\_file)

count = 0

for emp in data:

print(data[emp])

if count == 0:

# Writing headers of CSV file

header = ['Type','Probability']

csv\_writer.writerow(header)

count += 1

# Writing data of CSV file

d = [emp,data[emp]]

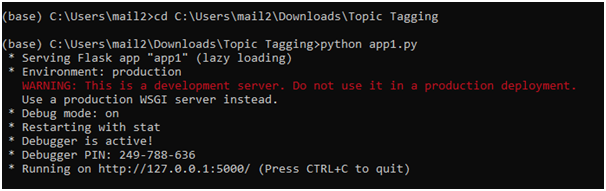
print(d)

csv\_writer.writerow(d)

data\_file.close()

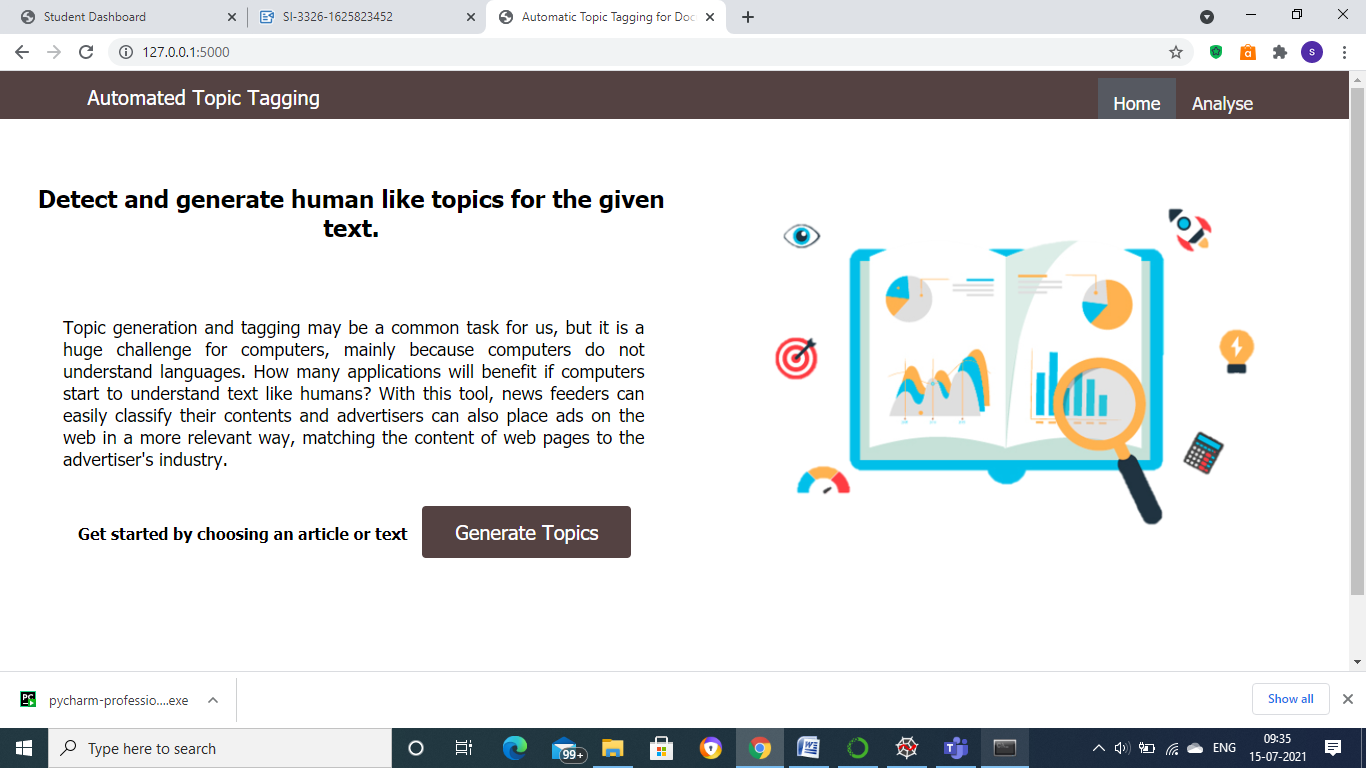
**Running of flask Application**

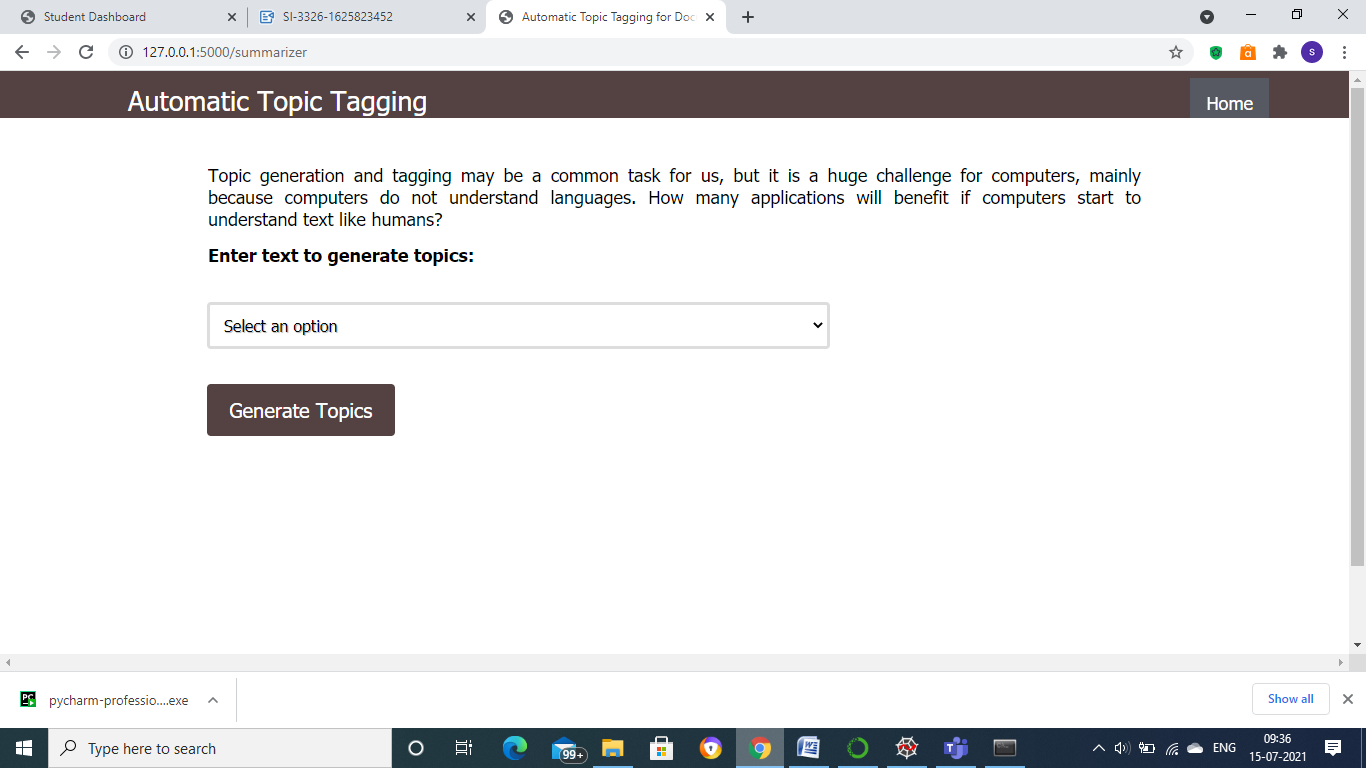
* Open anaconda prompt from start menu.
* Navigate to the folder where your app.py resides.
* Now type “python app.py” command.
* It will show the local host where your app is running on[**http://127.0.0.1.5000/**](http://127.0.0.1.5000/)
* Copy that local host URL and open that URL in browser. It does navigateyou to the where you can view your web page.

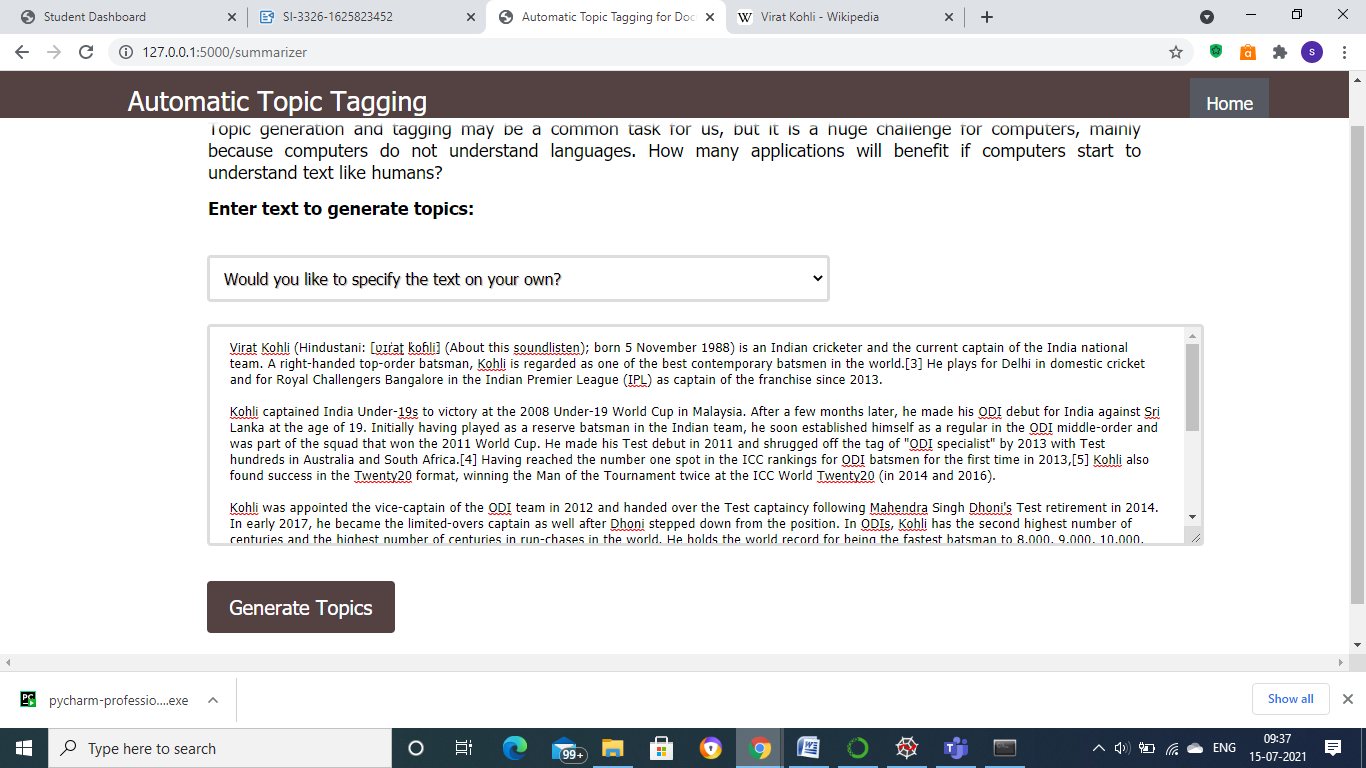


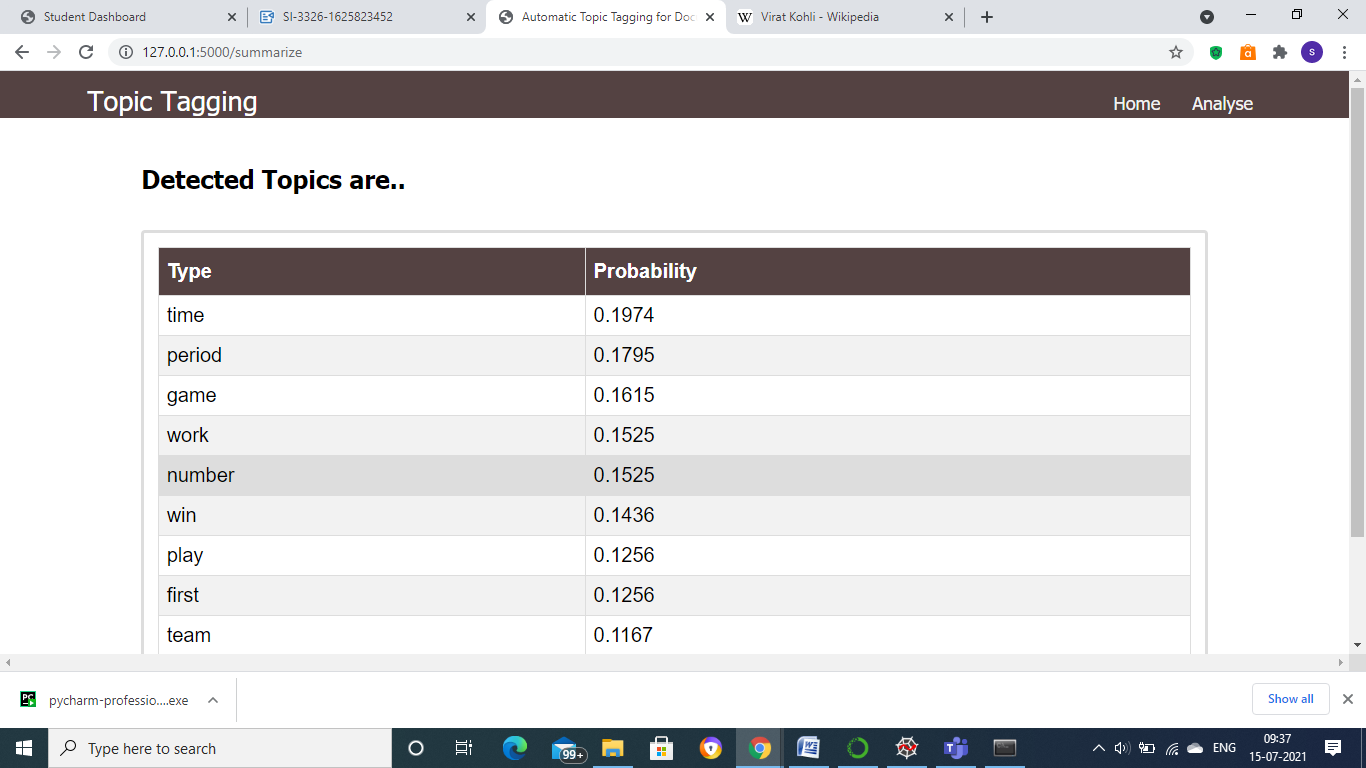
b. UI Output Screenshots:

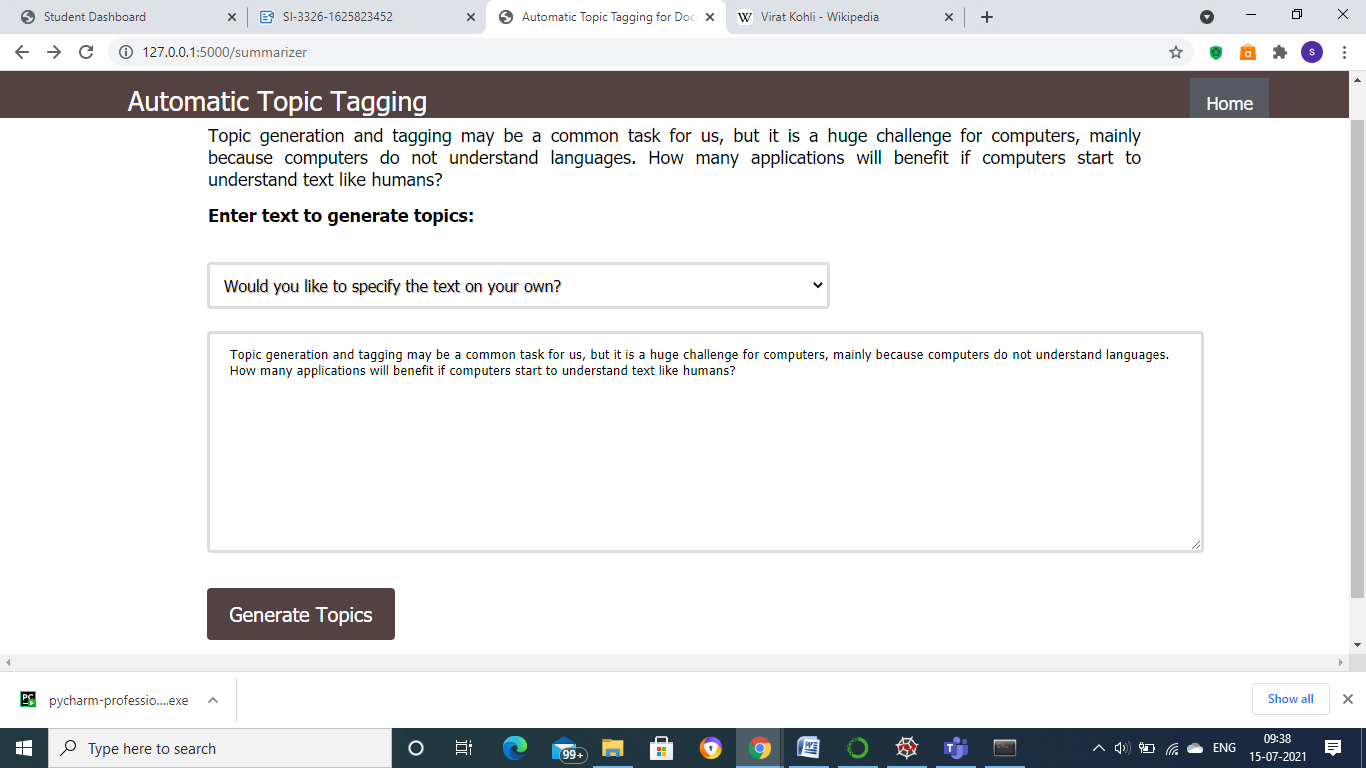
* UI look like:

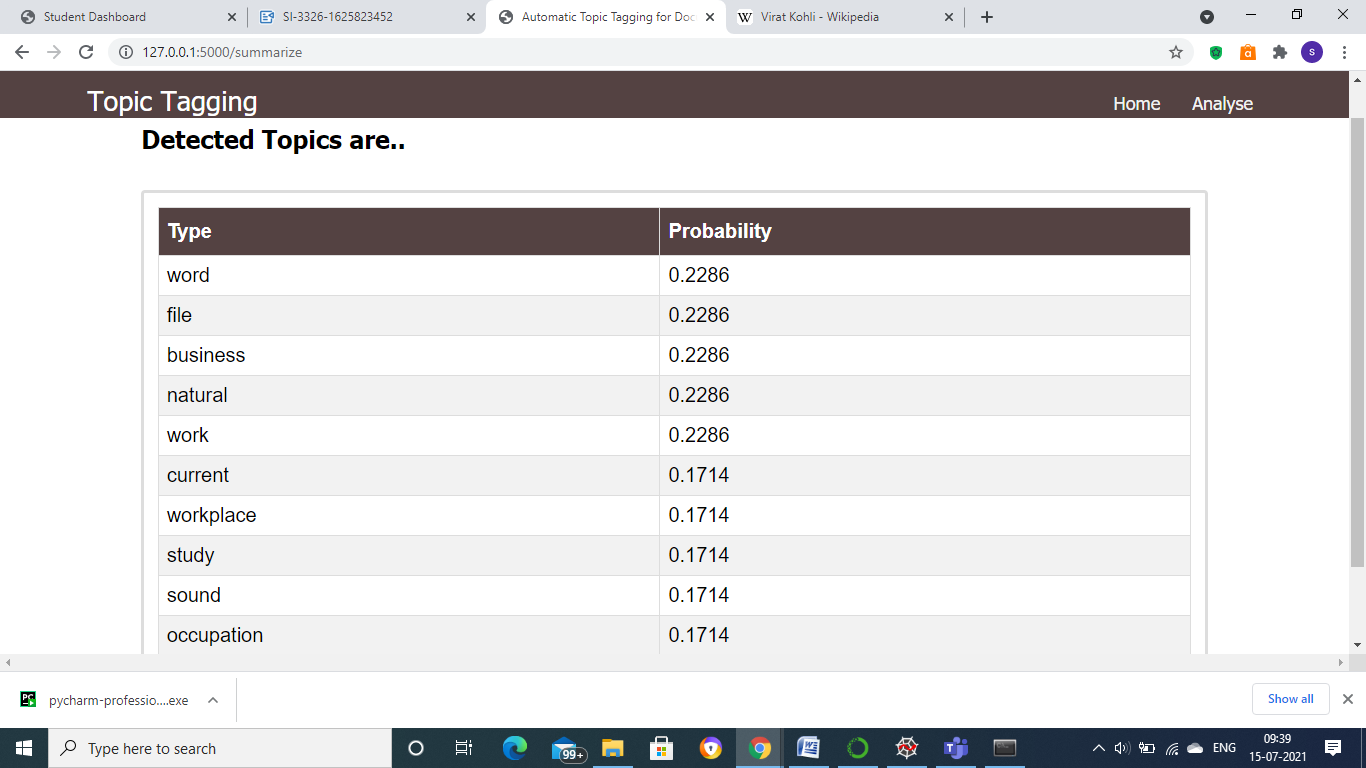












**........................**