Contents

[INTRODUCTION 2](#_Toc90079465)

[About Project 2](#_Toc90079466)

[Objective of the System 3](#_Toc90079467)

[SYSTEM SPECIFICATION 4](#_Toc90079468)

[Hardware Requirements:- 4](#_Toc90079469)

[Software Requirements: 5](#_Toc90079470)

[SCRAPING, PARSING,AND CRAWLING THE WEBPAGE 5](#_Toc90079471)

[Crawling 5](#_Toc90079472)

[Indexing 5](#_Toc90079473)

[Retrieval or Searching 5](#_Toc90079474)

[TYPES OF WEB CRAWLER 6](#_Toc90079475)

[SYSTEM DESIGN AND WORKING 7](#_Toc90079476)

[Design Steps 7](#_Toc90079477)

[Step 1 — Creating a Basic Scraper 8](#_Toc90079478)

[Step 2 — Extracting Data from a Page 8](#_Toc90079479)

[USE CASE DIAGRAMS 11](#_Toc90079480)

[SYSTEM TESTING AND IMPLEMENTATION 12](#_Toc90079481)

[OUTPUT RESULTS 14](#_Toc90079482)

[LEGAL CONSIDERATION 16](#_Toc90079483)

[CONCLUSION 17](#_Toc90079484)

[REFERENCES 18](#_Toc90079485)

## INTRODUCTION

As the data generated on the internet exponentially increases, developing guided data collection methods become more and more essential to the research process. This paper proposes an

approach to building a self-guiding web-crawler to collect data specifically from extremist websites. The guidance component of the web-crawler is achieved through the use of sentiment-based classification rules which allow the crawler to make decisions on the content of the webpage it downloads. First, content from 2,500webpages was collected for each of the four different sentiment based classes: extremist webpages, news sources discussing extremism, anti-extremist webpages, and webpages that have nothing to do with extremism. Then Parts of Speech (POS) tagging was used to find the most frequent keywords in these pages. Utilizing sentiment software, the sentiment value of each keyword was then calculated for each page. This was then used as the basis for developing a decision tree that could effectively discern which

class a particular page would fall into. The tree developed as a result of this process showed an 80% success rate on differentiating between the four classes and a 92% success rate at determining specifically extremist pages. This decision tree was then applied to a randomly selected sample of pages for each class. The results from the secondary test showed similar results to the primary analysis and show promise for future studies utilizing this framework.

## Objective of the System

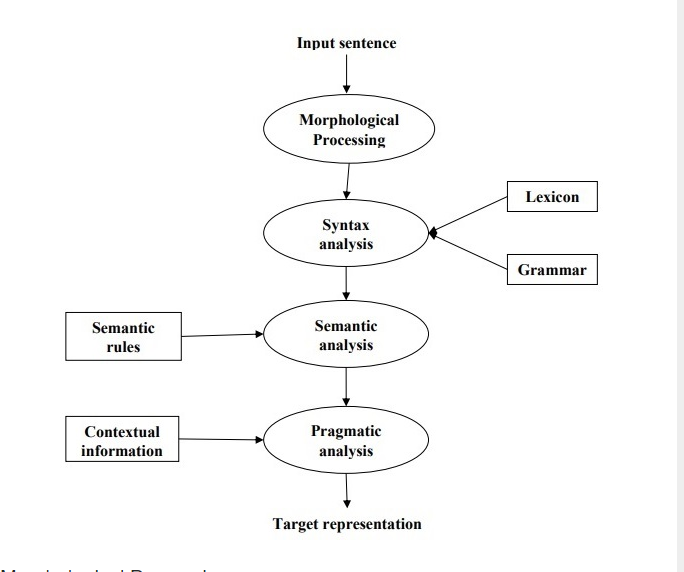
The main purpose of this project is to apply NLP data science skills, knowledge, and techniques to solve problems in data science NLP projects with a focus on web crawler and content extraction from webpages and applying NLP tasks in Python

**Literature review of the NLP task**

A language developed naturally without planning and modeling is considered a

Natural Language. A well known example is the English language. A Natural Language

could be considered as the counterpart of a computing code (LYONS, 1993).NLP research dates back to the late 1940s with Machine translation (MT) being said to be the first computer-based application related to natural language. It was Weaver and Booth who started one of the earliest MT projects in 1946, on computer translation based on expertise in breaking enemy codes during World War II. However, a general agreement was made that, Weaver’s memorandum of 1949 has brought the idea of MT to general notice and had inspired many projects(Joseph et al., 2016). NLP researchers are now developing next generation NLP systems that deal reasonably well with general text and account for a good portion of the variability and ambiguity of a language. Statistical approaches thrived in dealing with many generic problems in computational linguistics such as part-of-speech identification, word sense disambiguation, etc., and have become standard throughout NLP



## SYSTEM SPECIFICATION

### Hardware Requirements:-

* Pentium-IV(Processor).
* 256 MB Ram
* 512 KB Cache Memory
* Hard disk 10 GB
* Microsoft Compatible 101 or more Key Boar

### Software Requirements:

* Operating System :           Windows
* Programming  language:   Python/Flask
* Front-End:                          HTML, CSS, JAVASCRIPT
* Back-End:                           Mongo Db(Online)
* Web Server:                       PyMongo

## SCRAPING, PARSING,AND CRAWLING THE WEBPAGE

A **Web crawler**, sometimes called a **spider** or **spiderbot** and often shortened to **crawler**, is an [Internet bot](https://en.wikipedia.org/wiki/Internet_bot) that systematically browses the [World Wide Web](https://en.wikipedia.org/wiki/World_Wide_Web), typically operated by search engines for the purpose of [Web indexing](https://en.wikipedia.org/wiki/Web_indexing).

[Web search engines](https://en.wikipedia.org/wiki/Web_search_engine) and some other [websites](https://en.wikipedia.org/wiki/Website) use Web crawling or spidering [software](https://en.wikipedia.org/wiki/Software) to update their [web content](https://en.wikipedia.org/wiki/Web_content) or indices of other sites' web content. Web crawlers copy pages for processing by a search engine, which [indexes](https://en.wikipedia.org/wiki/Index_(search_engine)) the downloaded pages so that users can search more efficiently.

### Crawling

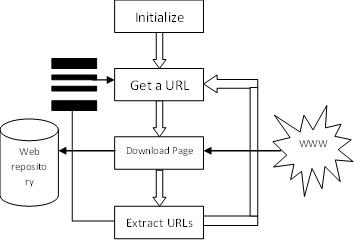
Web Crawlers such as “spider”, “bot” etc. are used for crawling. Web Crawlers are a program or automated script which browses the World Wide Web in a methodical, automated manner. Crawling is never ending process. Crawler module extracts data and key information from each page. Web crawlers can copy all the pages and store them in page repository and visit for later processing. When large parts of the Internet were essentially invisible to search engines – “deep web” (this is rare now). TOR-hosted websites remain unindexed by Google and are only accessible by connecting to the TOR network and knowing the address.

### Indexing

It refers to various methods for indexing the contents of a website or of the Internet as a whole. Search engines usually use keywords and metadata to provide a more useful vocabulary for Internet or onsite searching Search engine indexing collects, parses, and stores data to facilitate fast and accurate information retrieval. The purpose of storing an index is to optimize speed and performance in finding relevant documents for a search query. Indexed documents are then stored in databases.

### Retrieval or Searching

When user input any text for searching, that keyword is searched from database where condensed summary of web pages are stored After finding entered keyword relative webpages, page rank algorithm is implemented on those web pages. The page having highest rank is showed as first website of search result page.



### TYPES OF WEB CRAWLER

1. Distributed Crawler:

Many crawlers are working to distribute in the process of web crawling, in order to have the most coverage of the web. A central server manages the communication and synchronization of the nodes, as it is geographically distributed. It basically uses Page rank algorithm for its increased efficiency and quality search. The benefit of distributed web crawler is that it is robust against system crashes and other events, and can be adapted to various crawling applications

1. Parallel Crawler

Multiple crawlers are often run in parallel, which are referred as Parallel crawlers. The Parallel crawlers depend on Page freshness and Page Selection .A Parallel crawler can be on local network or be distributed at geographically distant locations. Parallelization of crawling system is very vital from the point of view of downloading documents in a reasonable amount of time

1. Focused Web Crawler

It tries to download pages that are related to each other. It collects documents which are specific and relevant to the given topic. It is also known as a Topic Crawler because of its way of working. It determines how far the given page is relevant to the particular topic and how to proceed forward. The benefits of focused web crawler is that it is economically feasible in terms of hardware and network resources It can reduce the amount of network traffic and downloads. Vertical search engine uses focused web crawler

1. Incremental Crawler

An incremental crawler incrementally refreshes the existing collection of pages by visiting them frequently; based upon the estimate as to how often pages change. It also exchanges less important pages by new and more important pages. It resolves the problem of the freshness of the pages. The benefit of incremental crawler is that only the valuable data is provided to the user, thus network bandwidth is saved and data enrichment is achieved.

## SYSTEM DESIGN AND WORKING

Design is the first step in the development phase for any techniques and principles for the purpose of defining a device, a process, or system in sufficient detail to permit its physical realization.  
Once the software requirements have been analysed and specified, the software design involves three technical activities – design, coding, implementation, and testing that are required to build and verify the software.  
 I have designed this system using python and flask and included nltk library for natural language processing. Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions.

The design activities are of main importance in this phase, because in this activity, decisions ultimately affecting the success of the software implementation and its ease of maintenance are made. These decisions have the final bearing upon reliability and maintainability of the system. Design is the only way to accurately translate the customer’s requirements into finished software or a system.  
Design is the place where quality is fostered in development. Software design is a process through which requirements are translated into a representation of software. Software design is conducted in two steps. Preliminary design is concerned with the transformation of requirements into data.

### Design Steps

### Step 1 — Creating a Basic Scraper

Scraping is a two step process:

1. You systematically find and download web pages.For our case we used “https://stackoverflow.com/questions/”
2. You take those web pages and extract information from them

Scrapy is one of the most popular and powerful Python scraping libraries; it takes a “batteries included” approach to scraping, meaning that it handles a lot of the common functionality that all scrapers need so developers don’t have to reinvent the wheel each time. It makes scraping a quick and fun process!.

Scrapy, like most Python packages, is on PyPI (also known as pip). PyPI, the Python Package Index, is a community-owned repository of all published Python software.

### Step 2 — Extracting Data from a Page

If you look at [the page we want to scrape](http://brickset.com/sets/year-2016), you’ll see it has the following structure:

* There’s a header that’s present on every page.
* There’s some top-level search data, including the number of matches, what we’re searching for, and the breadcrumbs for the site.
* Then there are the sets themselves, displayed in what looks like a table or ordered list. Each set has a similar format.

When writing a scraper, it’s a good idea to look at the source of the HTML file and familiarize yourself with the structure. So here it is, with some things removed for readability.

**Crawler.py**

import requests

from bs4 import BeautifulSoup

import pandas as pd

letter = []

class Sentiment:

def get\_soup(url):

r = requests.get('https://mail.google.com/mail/u/0/#inbox')

soup = BeautifulSoup(r.text, 'html.parser')

return soup

def get\_reviews(soup):

reviews = soup.find\_all('div', {'data-hook': 'review'})

try:

for item in reviews:

review = {

'Category': soup.title.text.replace('https://mail.google.com/mail:title:', '').strip(),

'Message': item.find('a', {'data-hook': 'message'}).text.strip(),

}

letter.append(review)

except:

pass

for x in range(10):

soup = get\_soup(f'https://mail.google.com/mail/u/0/#inbox={x}')

print(f'Getting page: {x}')

get\_reviews(soup)

print(len(letter))

if not soup.find('li', {'class': 'a-disabled a-last'}):

pass

else:

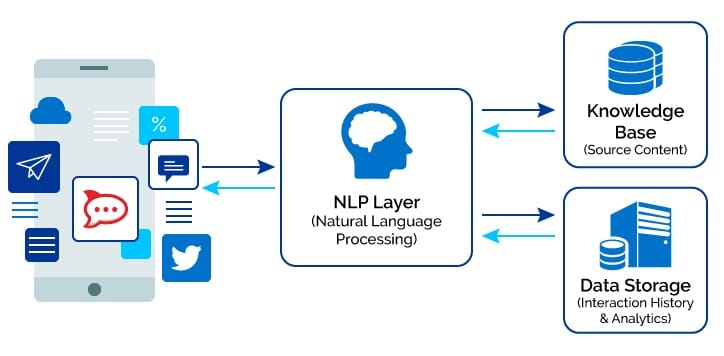
break

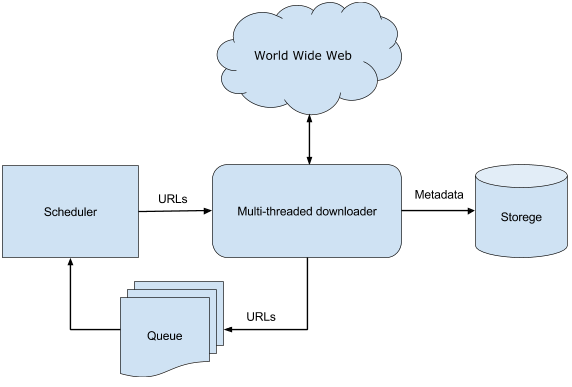
df = pd.DataFrame(letter)

print(df.head())

df.to\_csv(r'./data.csv', index=None)

### USE CASE DIAGRAMS

****



## SYSTEM TESTING AND IMPLEMENTATION

Testing is a process of executing a program with the indent of finding an error. Testing is a crucial element of software quality assurance and presents ultimate review of specification, design and coding.  
System Testing is an important phase. Testing represents an interesting anomaly for the software. Thus a series of testing are performed for the proposed system before the system is ready for user acceptance testing.  
A good test case is one that has a high probability of finding an as undiscovered error. A successful test is one that uncovers an as undiscovered error.

NLP is a subfield of artificial intelligence and is concerned with interactions between computers and human languages. NLP is the process of analysing, understanding, and deriving meaning from human languages for computers.

in the testing of this system ,i installed it in a separate machine and run the project and the following outcomes evolved.

* i had to install dependances in the machine for the system to send requests to the Pymongo server hosted online.
* i installed requests using command prompt

**Testing Objectives:**  
 1. Testing is a process of executing a program with the intent of finding an error  
 2. A good test case is one that has a probability of finding an as yet undiscovered error  
 3. A successful test is one that uncovers an undiscovered error

**Testing Principles**

* All tests should be traceable to end user requirements
* Tests should be planned long before testing begins.
* Testing should begin on a small scale and progress towards testing in large.
* Exhaustive testing is not possible.
* To be most effective testing should be conducted by a independent third party

The primary objective for test case design is to derive a set of tests that has the highest livelihood for uncovering defects in software. To accomplish this objective two different categories of test case design techniques are used. They are  
 1. White box testing.  
 2. Black box testing.

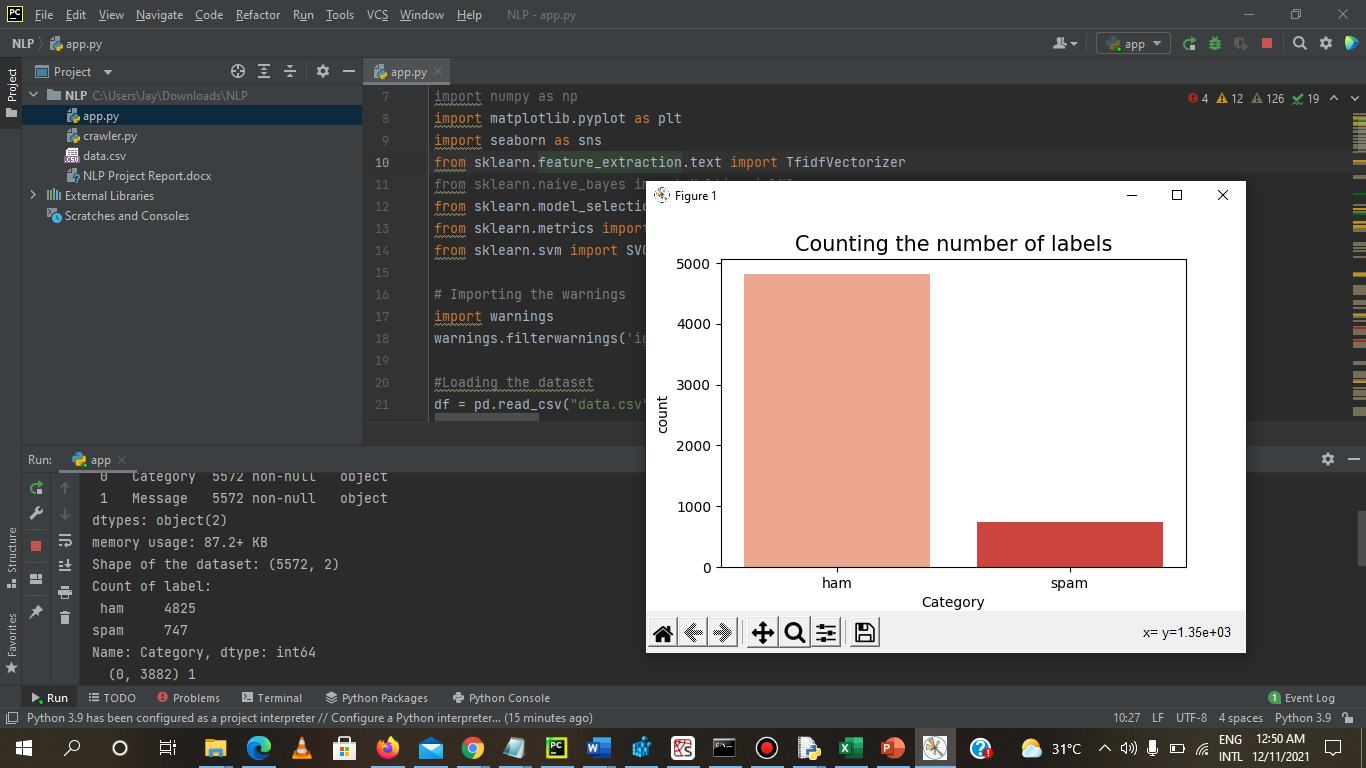
**Testing strategies**

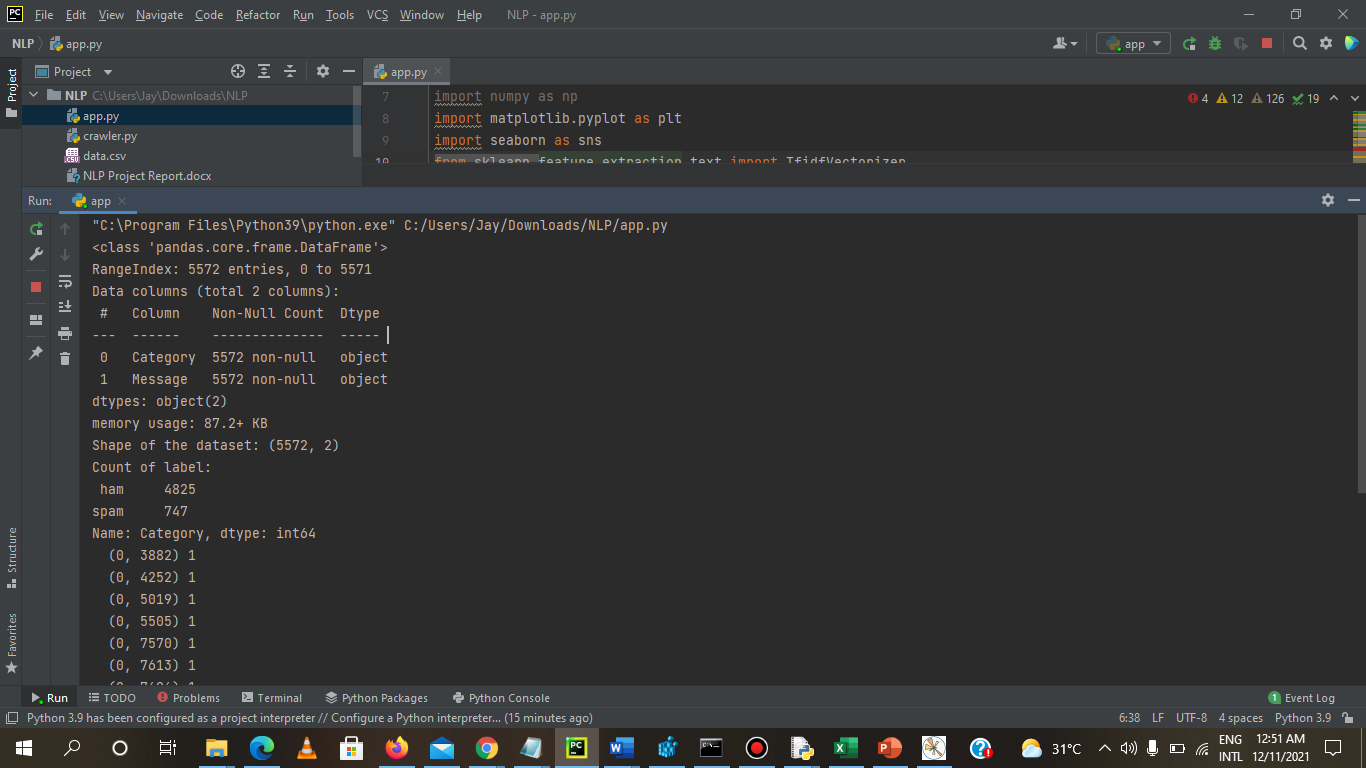
A strategy for software testing must accommodate low-level tests that are necessary to verify that all small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements.  
**Testing fundamentals**  
Testing is a process of executing program with the intent of finding error. A good test case is one that has high probability of finding an undiscovered error. If testing is conducted successfully it uncovers the errors in the software.   
  
**Testing Information flow**  
Information flow for testing flows the pattern. Two class of input provided to test the process. The software configuration includes a software requirements specification, a design specification and source code.  
  
**Unit testing**  
Unit testing is essential for the verification of the code produced during the coding phase and hence the goal is to test the internal logic of the modules. Using the detailed design description as a guide, important paths are tested to uncover errors with in the boundary of the modules. These tests were carried out during the programming stage itself. All units of Vienna SQL were successfully tested.

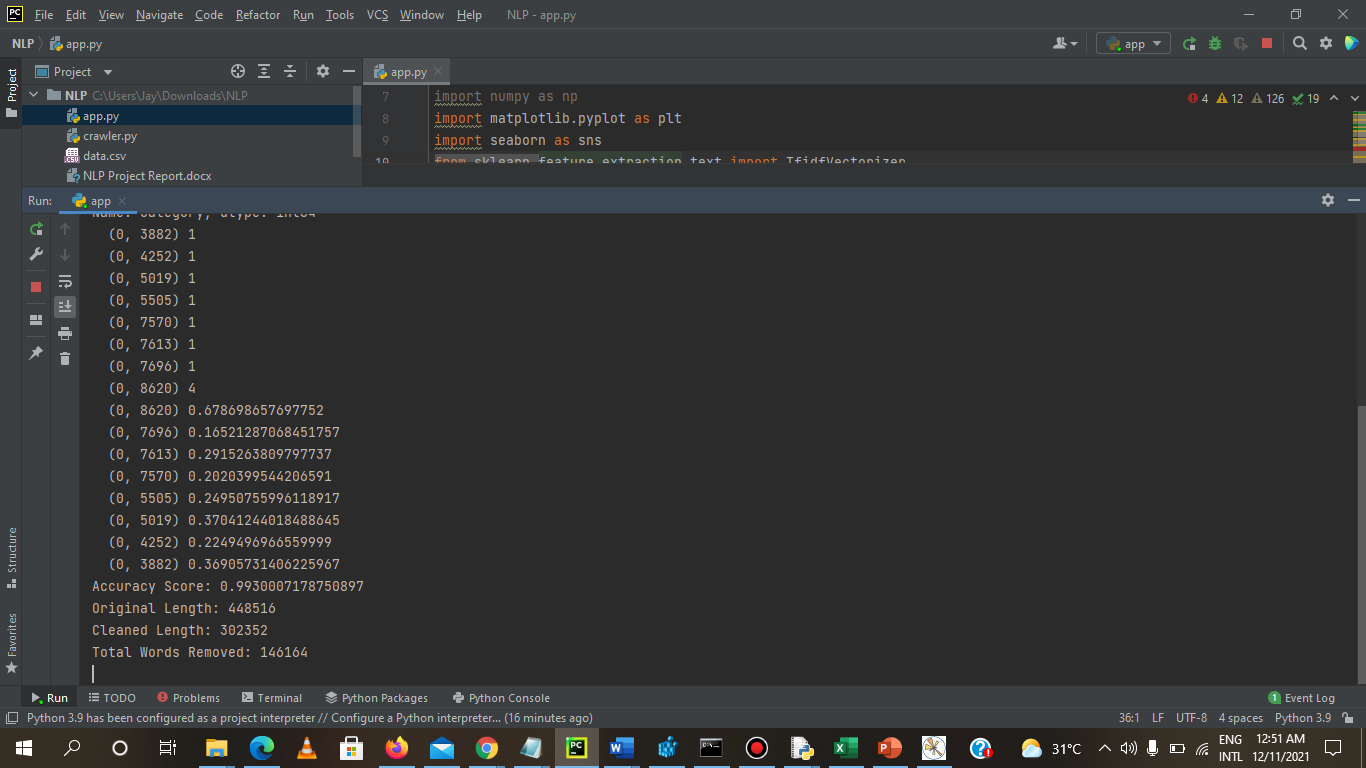
**Integration testing**   
Integration testing focuses on unit tested modules and build the program structure that is dictated by the design phase.

**System testing**  
System testing tests the integration of each module in the system. It also tests to find discrepancies between the system and it’s original objective, current specification and system documentation. The primary concern is the compatibility of individual modules. Entire system is working properly or not will be tested here, and specified path ODBC connection will correct or not, and giving output or not are tested here these verifications and validations are done by giving input values to the system and by comparing with expected output. Top-down testing implementing here.  
**Acceptance Testing**  
This testing is done to verify the readiness of the system for the implementation. Acceptance testing begins when the system is complete. Its purpose is to provide the end user with the confidence that the system is ready for use. It involves planning and execution of functional tests, performance tests and stress tests in order to demonstrate that the implemented system satisfies its requirements.

## OUTPUT RESULTS







## LEGAL CONSIDERATION

A software development agreement is between a customer and a developer, by which the customer contracts for the developer to create and deliver a specified piece of software. The following is a gives a non-exhaustive overview of five important issues that this kind of contract should address.

**1. Scope of Work**

Scope of work encompasses the project’s objectives and is the most important part of the agreement. It should address the time frame, tasks, deliverables, quality of work, fees and payment schedule. It also determines the specifications of a project including the coding language used and the precise technical functions of the software

**2.Representations and Warranties**

The developer will make various representations to the customer regarding the performance of the product and the accuracy and completeness of the documentation. Warranties are used to allocate risk between the parties to the customer and the developer.

**3.Ownership of Intellectual Property Rights**

Rights in software code and other work product created under a software development agreement are typically protected under copyright. Rights in software may also be protected under patent law for software-implemented processes and devices, and trade secrets for the software’s source code and related confidential documents and materials.

**4.Open-Source Software**

The developer will, invariably, use open-source software. Open-source software is computer software that has the source code made available to the public with a license from the copyright holder. It permits users to study, change, and distribute the software to anyone and for any purpose. The client may seek a full list of all open source software used by the developer. It is important to seek warranties from the developer that the developer has read, understood and complied with the requirements for using that open-source software

.

## 

## CONCLUSION

The system was designed in such a way that future modifications can be done easily. The following conclusions can be deduced from the development of the project.

* Automation of the entire system improves the efficiency
* It provides a friendly graphical user interface which proves to be much better and retrieves searches without buffering
* It effectively overcomes the delay in rendering results.
* The System has adequate scope for modification in future if it is necessary.

## REFERENCES

Berners-Lee, Tim, “The World Wide Web: Past, Present and Future”, MIT USA, Aug 1996, available at: http://www.w3.org/People/Berners-Lee/1996/ppf.html.

Maurice de Kunder, “Size of the World Wide Web”, Available at: http://www.worldwidewebsize.com (accessed on May 5, 2012)

T. Berners-Lee, R. Cailliau, J. Groff, and B. Pollermann. World-Wide Web: The Information Universe. Electronic Networking: Research, Applications and Policy, 1(2): pp. 74–82, 1992. URL http://citeseer.ist.psu.edu/bernerslee92worldwide.html.

Natural Language Processing - Introduction. (2021). Tutorialspoint.com. <https://www.tutorialspoint.com/natural_language_processing/natural_language_processing_introduction.htm>

T. Berners-Lee. W3C, Mar. 2008. URL <http://www.w3.org/>.

Joseph, S. R., Kutlwano Sedimo, Freeson Kaniwa, & Keletso Letsholo. (2016, March). Natural Language Processing: A Review. ResearchGate; unknown. <https://www.researchgate.net/publication/309210149_Natural_Language_Processing_A_Review>

LYONS, J. Language, Linguistic Society of America, v. 69, n. 4, p. 825–828, 1993

‌

‌

‌