File search engine/crawler

Project submitted to the

SRM University – AP, Andhra Pradesh

for the partial fulfillment of the requirements to award the degree of

Bachelor of Technology/Master of Technology

In

Computer Science and Engineering

School of Engineering and Sciences

Submitted by

GOLLAPALLI GEETHA SIVA SRINIVAS

(AP20110010316)

Under the Guidance of

(Dr.Shubham Gupta)

SRM University–AP

Neerukonda, Mangalagiri, Guntur

Andhra Pradesh – 522 240

[DECEMBER, 2022]

Certificate

Date: 20-Dec-22

This is to certify that the work present in this Project entitled "**FILE SEARCH ENGINE/CRAWLER**" has been carried out by **DOPPALAPUDI KALYAN KUMAR** under my/our supervision. The work is genuine, original, and suitable for submission to the SRM University – AP for the award of Bachelor of Technology/Master of Technology in **School of Engineering and Sciences**.

Supervisor

(Signature)

Prof. / Dr. Shubham Gupta

Assistant Professor.

Table of Contents

Abstract

File search engine/crawler There are hundreds of web search engines or web crawlers which are out on the internet. Google is one of the best search engines. As many people show interest on google it is the best search engine/crawler. Internet searcher is the mainstream term for a data recovery Incident response (IR) framework. While analysts and designers take a more extensive perspective of IR frameworks, buyers consider them more as far as what they need the frameworks to do — to be specific, hunt the Web, or an intranet, or a database. Current search engine result pages exhibit a very complex structure. They contain organic results, ads (text ads, shopping ads), verticals, direct answers, and knowledge graph results. The search engine market is dominated by Google, there is a need to promote user information literacy to create a more solid foundation for user trust in search engines. A very simplified version of the file search engine project built with PySimpleGUI except using a list box. When an item is clicked on, the program will attempt to open with the default program. We are actually aiming to incorporate the best possible local file search engine with the goal of getting the correct file in our File search engine project.

List of Figures

1. Introduction

A Web crawler is an Internet bot that routinely browses the World Wide Web and is often run by search engines for the purpose of Web indexing. It is also known as a spider, spiderbot, or just a crawler (web spidering).

To update its web content or indices of other sites' web content, web search engines and some other websites employ Web crawling or spidering software. Web crawlers copy pages for a search engine to process and index, enabling users to conduct more effective searches.

Crawlers frequently visit sites without being asked and consume resources on systems they visit. When enormous collections of pages are accessed, issues of schedule, load, and "politeness" come into play.

Crawlers visit websites and read their pages and other information to create entries for a search engine's index. The primary purpose of a web crawler is to provide users with a comprehensive and up-to-date index of all available online content.

Search engines like Google, Bing, and Yahoo use crawlers to properly index downloaded pages so that users can find them faster and more efficiently when searching. Without web crawlers, there would be nothing to tell them that your website has new and fresh content. Sitemaps also can play a part in that process. So web crawlers, for the most part, are a good thing.

1.1 File-Search-Engine

A file search engine project is a type of software project that uses computer algorithms to search through large amounts of data for a particular file or set of files. The goal of a file search engine is to quickly and accurately locate the desired file or files in the shortest possible time.

The project typically begins with the development of a file indexing algorithm that is used to scan and analyze the data. The indexing algorithm is used to create a list of the files in the data set. This list is then used to create a searchable database that the search engine can use to quickly locate the desired files.

There are mechanisms for public websites that do not want to be crawled to let the crawling agent know. A robots.txt file, for instance, can instruct search engine bots to index only specific pages of a website or none at all.

Even the largest crawlers cannot provide a full index due to the sheer volume of Internet pages. In the early years of the World Wide Web, prior to 2000, search engines struggled to provide pertinent search results. Relevant results are now provided very immediately.

1.1.1 Features

A simple GUI based utility to index and search local files. So... perhaps the GUI is a bit unnecessary, but why not.

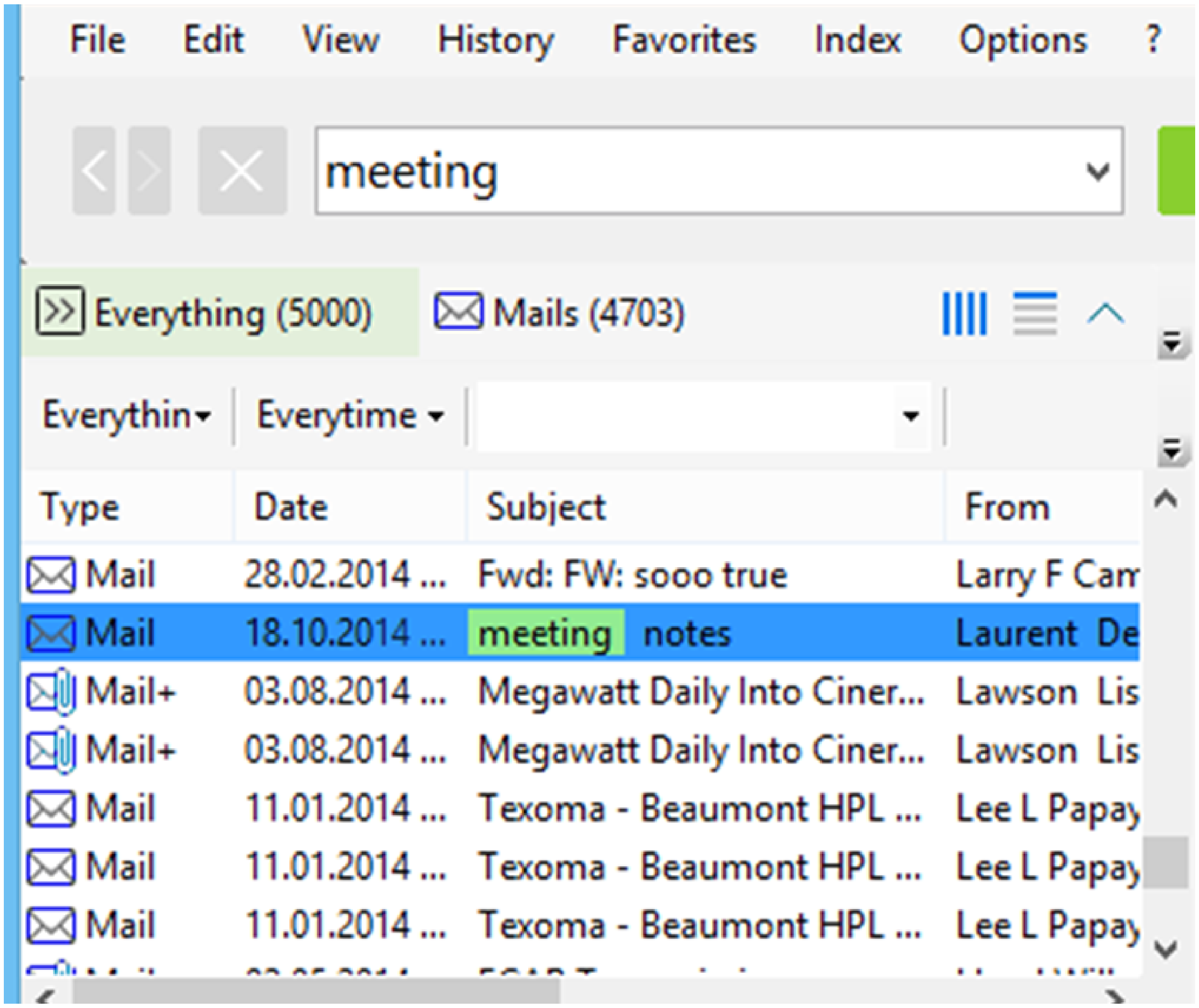
Search options include: 'contains', 'startswith', and 'endswith' Re-index saves a local copy of the index for quick re-use. This makes subsequent searches nearly instant. Scope the domain of your search by browsing to a higher level root directory using the Browse button. Search results are both printed to screen and exported to a text file in the working directory. Requirements PySimpleGUI for the GUI pip install PySimpleGUI.

1.2 Approach

Once the search engine is built, it can be tested and refined to ensure it is accurate and efficient. The search engine is also often used to provide an interface for users to make their searches more refined and specific.

The project also involves making sure the search engine works with different types of file formats, such as text, images, and videos. This is important so that users can find the right file quickly and accurately.

Finally, the project also involves monitoring and maintaining the search engine to ensure it is working properly and remains up to date. This includes periodically updating the search engine with the latest data and ensuring it is running smoothly.



Type	Date	Subject	From
Mail	28.02.2014 ...	Fwd: FW: sooo true	Larry F Cam
Mail	18.10.2014 ...	meeting notes	Laurent De
Mail+	03.08.2014 ...	Megawatt Daily Into Ciner...	Lawson Lis
Mail+	03.08.2014 ...	Megawatt Daily Into Ciner...	Lawson Lis
Mail	11.01.2014 ...	Texoma - Beaumont HPL ...	Lee L Papay
Mail	11.01.2014 ...	Texoma - Beaumont HPL ...	Lee L Papay
Mail	11.01.2014 ...	Texoma - Beaumont HPL ...	Lee L Papay

Fig1.Main approach

2. Methodology

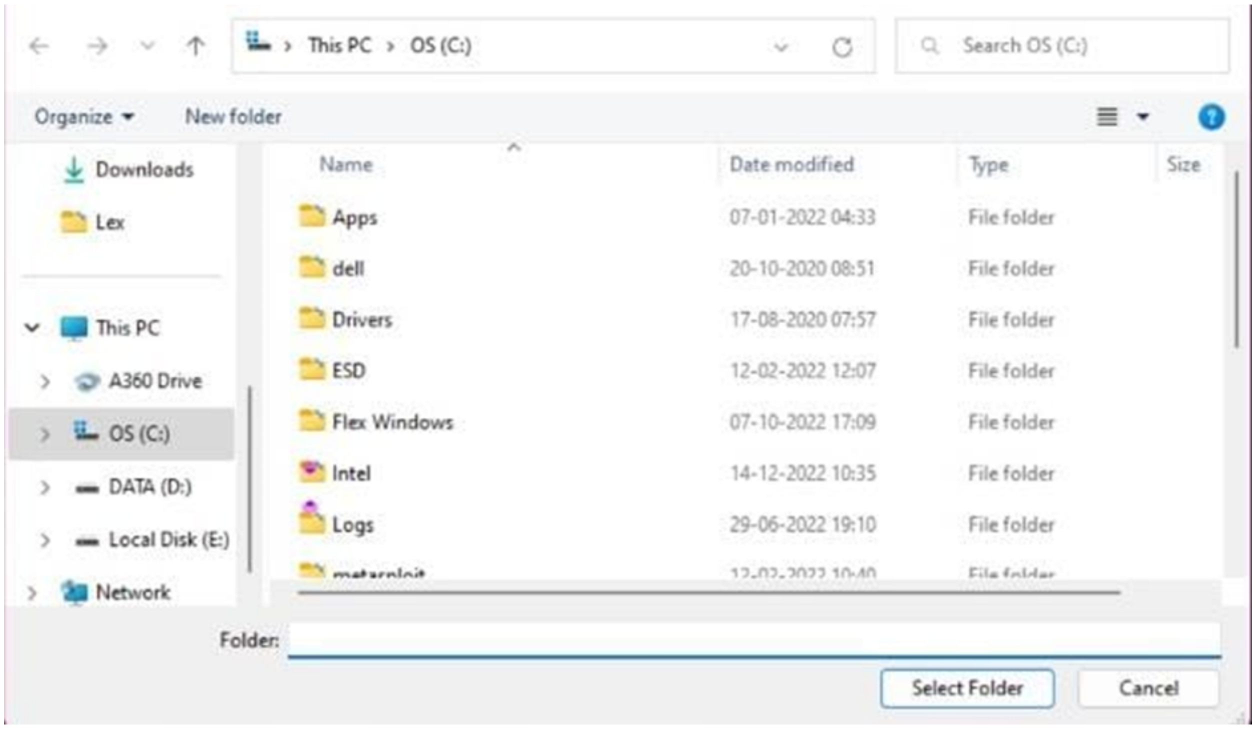
CODE:

"""

```
 A very simplified version of the file search engine project built with PySimpleGUI

   except using a listbox. When an item is click on, the program will attempt to open
with the

 default program.

"""

import PySimpleGUI as sg

import os



def search(values, window):

    """Perform a search based on term and type"""

    global results

      # reset the results list

    results.clear()

    window['-RESULTS-'].update(values=results)

    window['-INFO-'].update(value='Searching for matches...')



      # search for term and save new results

      for root, _, files in os.walk(values['-PATH-']):

        for file in files:

            if values['-ENDSWITH-'] and file.lower().endswith(values['-TERM-'].lower()):

                results.append(f'{root}\\{file}'.replace('\\', '/'))

                window['-RESULTS-'].update(results)

            if values['-STARTSWITH-'] and file.lower().startswith(values['-TERM-'].lower()):

                results.append(f'{root}\\{file}'.replace('\\', '/'))

                window['-RESULTS-'].update(results)
            if values['-CONTAINS-'] and values['-TERM-'].lower() in file.lower():
                results.append(f'{root}\\{file}'.replace('\\', '/'))
                window['-RESULTS-'].update(results)
     window['-INFO-'].update('Enter a search term and press `Search`')
    sg.PopupOK('Finished!')


def open_file(file_name):
    """Attempt to open the file with the default program"""
     # probably should add error handling here for when a default program cannot be
found.
    os.system(file_name)


# create the main file search window
results = []
sg.change_look_and_feel('Black')
layout = [
    [sg.Text('Search Term', size=(11, 1)), sg.Input('', size=(40, 1), key='-TERM-'),
     sg.Radio('Contains', group_id='search_type', size=(10, 1), default=True,
key='-CONTAINS-'),
     sg.Radio('StartsWith', group_id='search_type', size=(10, 1), key='-STARTSWITH-'),
     sg.Radio('EndsWith', group_id='search_type', size=(10, 1), key='-ENDSWITH-')],
    [sg.Text('Search Path', size=(11, 1)), sg.Input('/..', size=(40, 1), key='-PATH-'),
     sg.FolderBrowse(size=(10, 1), key='-BROWSE-'),
     sg.Button('Search', size=(10, 1), key='-SEARCH-')],
     [sg.Text('Enter a search term and press `Search`', key='-INFO-')],
  [sg.Listbox(values=results, size=(100, 28), enable_events=True, key='-RESULTS-')]]


window = sg.Window('File Search Engine', layout=layout, finalize=True,
return_keyboard_events=True)

window['-RESULTS-'].expand(expand_x=True, expand_y=True)


# main event loop

while True:

  event, values = window.read()

   if event is None:

    break

   if event == '-SEARCH-':

    search(values, window)

   if event == '-RESULTS-':

    file_name = values['-RESULTS-']

    if file_name:

      open_file(file_name[0])
```

2.1 Data Collection

The data files used in this project are the files that are present in the machine. We define a path for the search engine to search the requested files.



Name	Date modified	Type	Size
Apps	07-01-2022 04:33	File folder	
dell	20-10-2020 08:51	File folder	
Drivers	17-08-2020 07:57	File folder	
ESD	12-02-2022 12:07	File folder	
Flex Windows	07-10-2022 17:09	File folder	
Intel	14-12-2022 10:35	File folder	
Logs	29-06-2022 19:10	File folder	
metasploit	12-02-2022 10:40	File folder	

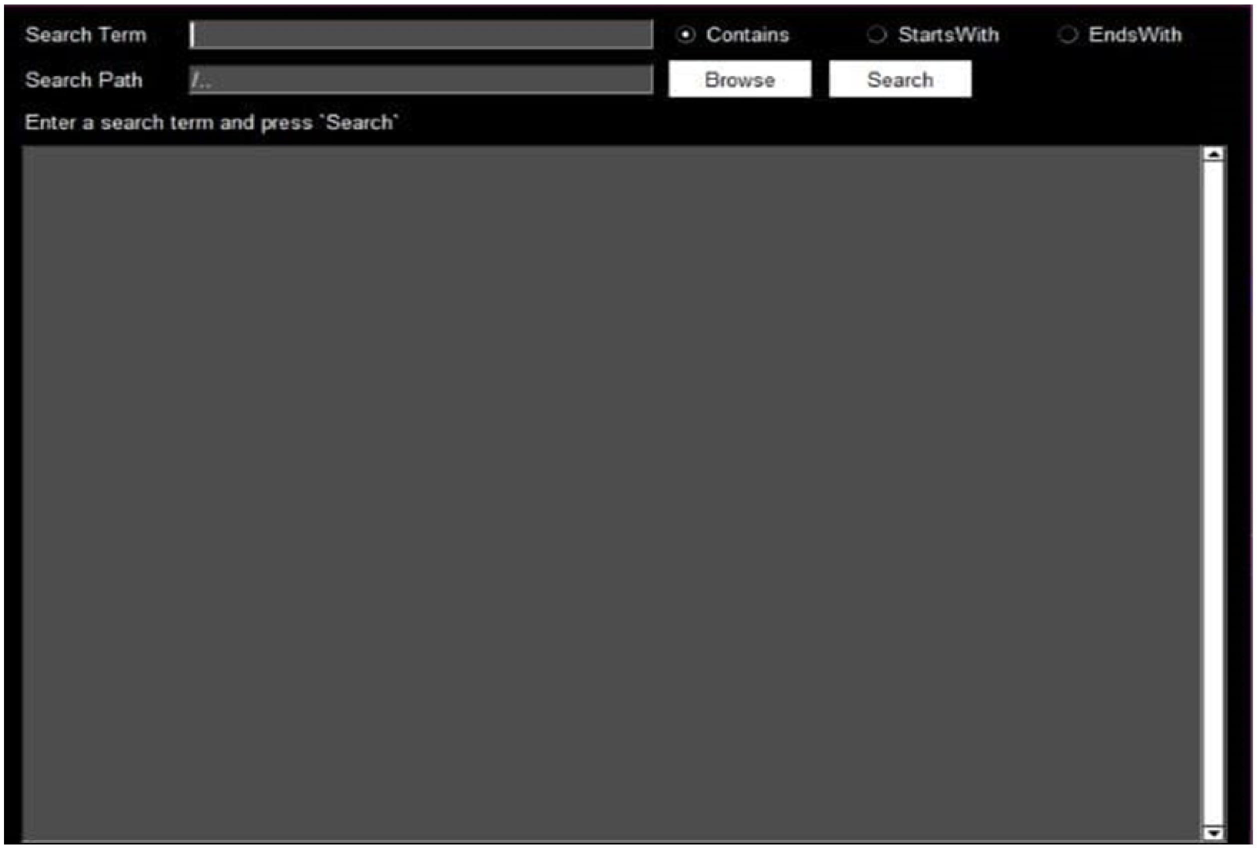
Fig2.Data Collection

2.1.1 Python Functions

We used packages like PySimpleGUI in this project. PySimpleGUI is a Python package that enables Python programmers of all levels to create GUIs.We specify our GUI window using a "layout" that contains widgets.

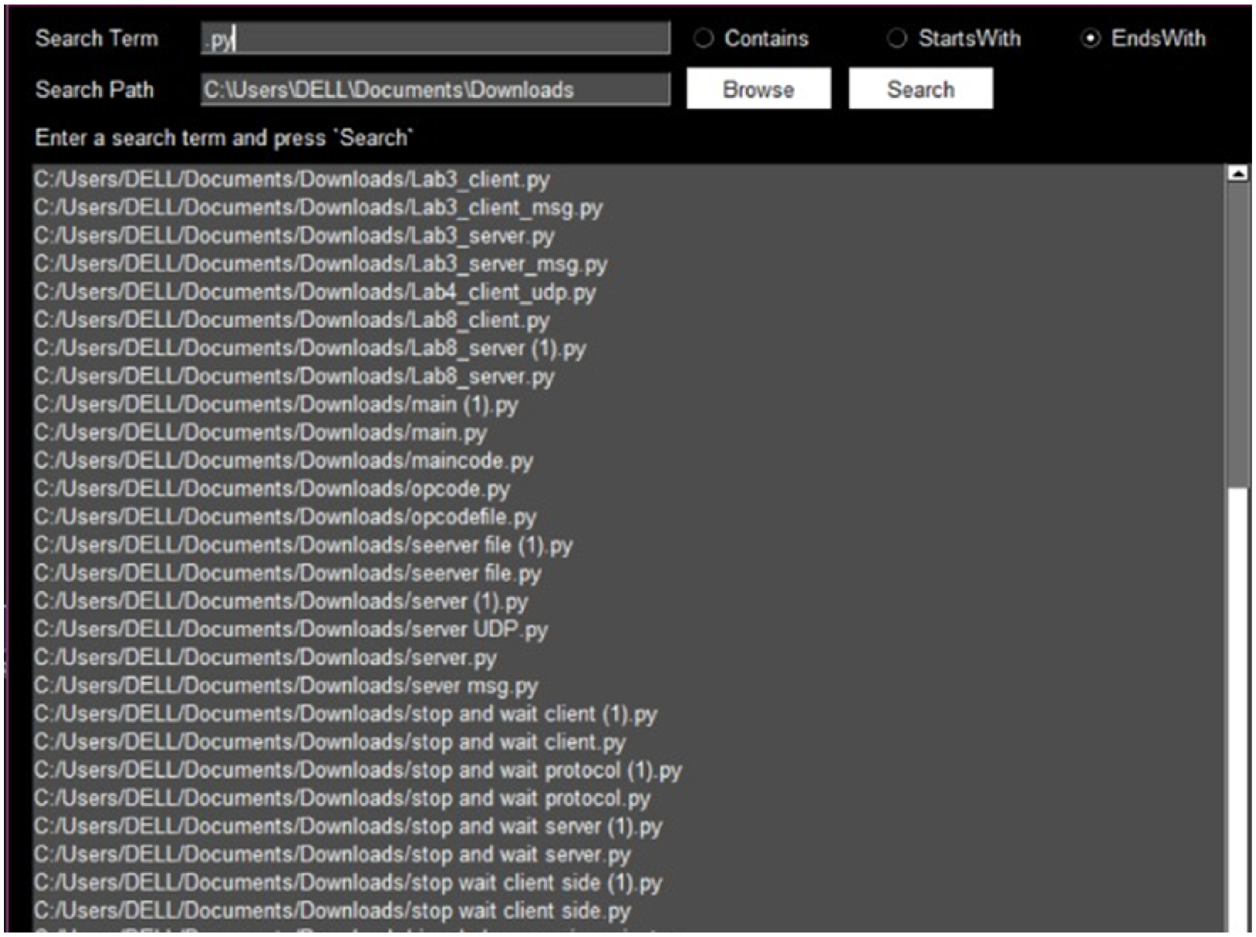
Python method walk () is used. It generates the file names in a directory.

RESULT:



Enter a search term and press `Search`

Layout



Search Term: .py [✓ EndsWith]

Search Path: C:\Users\DELL\Documents\Downloads

Enter a search term and press `Search`

```
C:/Users/DELL/Documents/Downloads/Lab3_client.py
C:/Users/DELL/Documents/Downloads/Lab3_client_msg.py
C:/Users/DELL/Documents/Downloads/Lab3_server.py
C:/Users/DELL/Documents/Downloads/Lab3_server_msg.py
C:/Users/DELL/Documents/Downloads/Lab4_client_udp.py
C:/Users/DELL/Documents/Downloads/Lab8_client.py
C:/Users/DELL/Documents/Downloads/Lab8_server (1).py
C:/Users/DELL/Documents/Downloads/Lab8_server.py
C:/Users/DELL/Documents/Downloads/main (1).py
C:/Users/DELL/Documents/Downloads/main.py
C:/Users/DELL/Documents/Downloads/maincode.py
C:/Users/DELL/Documents/Downloads/opcode.py
C:/Users/DELL/Documents/Downloads/opcodefile.py
C:/Users/DELL/Documents/Downloads/seerver file (1).py
C:/Users/DELL/Documents/Downloads/seerver file.py
C:/Users/DELL/Documents/Downloads/server (1).py
C:/Users/DELL/Documents/Downloads/server UDP.py
C:/Users/DELL/Documents/Downloads/server.py
C:/Users/DELL/Documents/Downloads/sever msg.py
C:/Users/DELL/Documents/Downloads/stop and wait client (1).py
C:/Users/DELL/Documents/Downloads/stop and wait client.py
C:/Users/DELL/Documents/Downloads/stop and wait protocol (1).py
C:/Users/DELL/Documents/Downloads/stop and wait protocol.py
C:/Users/DELL/Documents/Downloads/stop and wait server (1).py
C:/Users/DELL/Documents/Downloads/stop and wait server.py
C:/Users/DELL/Documents/Downloads/stop wait client side (1).py
C:/Users/DELL/Documents/Downloads/stop wait client side.py
```

Fig3.Output window

3. Concluding Remarks

When using a web search engine, a good way to test the spidering frequency is to use the cached copy that's made available on several web search engines. Web search engines often have to deal with online data. A search engine may be able to afford constructing a model offline on huge data sets. We are actually aiming to incorporate the best possible local file search engine with the goal of getting the correct file in our File search engine project and implementing them.

4. Future Work

We can able to build a local file search engine/crawler. By using some updated functions and concepts we can create a search engine like Google or bing or Yahoo. Also building bot which is a very effective web crawler that can index pages quickly and accurately. However, it does have some drawbacks. For example, Googlebot does not always crawl all the pages on a website.

References

1. Efthimiadis, E.N. and Hendry, D.G., 2005, August. Search engines and how students think they work. In Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval (pp. 595-596).

2. Butcher, Stefan, Charles LA Clarke, and Gordon V. Cormack. Information retrieval: Implementing and evaluating search engines. Mit Press, 2016.

3. Bergman, Ofer, et al. "Improved search engines and navigation preference in personal information management." ACM Transactions on Information Systems (TOIS) 26.4 (2008): 1-24.