SEARCH ENGINE

A PROJECT REPORT

Submitted by

SIBI CHAKKARAVARTHI S (2303811724321105)

in partial fulfillment of requirements for the award of the course

CGB1201 – JAVA PROGRAMMING

in

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM – 621 112

DECEMBER, 2024

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY (AUTONOMOUS)

SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this project report on **“SEARCH ENGINE”** is the bonafide work of **SIBI CHAKKARAVARTHI S (2303811724321105)** who carried out the project work during the academic year 2024 - 2025 under my supervision.

[illegible]

S. Geetha

Signature	**Signature**
Dr. T. AVUDAIAPPAN M.E.,Ph.D.,	Mrs. S. GEETHA M.E.,
HEAD OF THE DEPARTMENT,	**SUPERVISOR,**
Department of Artificial Intelligence,	Department of Artificial Intelligence,
K. Ramakrishnan College of Technology,	K. Ramakrishnan College of Technology,
Samayapuram, Trichy -621 112.	Samayapuram, Trichy -621 112.

Submitted for the viva-voce examination held on 3.12.24

S. Geetha

[illegible]

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION

I declare that the project report on “**SEARCH ENGINE**” is the result of original work done by me and best of my knowledge, similar work has not been submitted to “**ANNA UNIVERSITY CHENNAI**” for the requirement of Degree of **BACHELOR OF TECHNOLOGY**. This project report is submitted on the partial fulfillment of the requirement of the award of the **CGB1201 – JAVA PROGRAMMING.**

[illegible]

Signature

SIBI CHAKKARAVARTHI S

Place: Samayapuram

Date: 3/12/2024

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and indebtedness to our institution, **“K. Ramakrishnan College of Technology (Autonomous)”,** for providing us with the opportunity to do this project.

I extend our sincere acknowledgement and appreciation to the esteemed and honourable Chairman, **Dr. K. RAMAKRISHNAN, B.E.,** for having provided the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director, **Dr. S. KUPPUSAMY, MBA, Ph.D.,** for forwarding our project and offering an adequate duration to complete it.

I would like to thank **Dr. N. VASUDEVAN, M.TECH., Ph.D.,** Principal, who gave the opportunity to frame the project to full satisfaction.

I thank **Dr.T.AVUDAIAPPAN, M.E.,Ph.D**., Head of the Department of **ARTIFICIAL INTELLIGENCE AND DATA SCIENCE**, for providing her encouragement in pursuing this project.

I wish to convey our profound and heartfelt gratitude to our esteemed project guide **Mrs.S.GEETHA M.E**., Department of **ARTIFICIAL INTELLIGENCE AND DATA SCIENCE**, for her incalculable suggestions, creativity, assistance and patience, which motivated us to carry out this project.

I render our sincere thanks to the Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

VISION OF THE INSTITUTION

To serve the society by offering top-notch technical education on par with global standards.

MISSION OF THE INSTITUTION

- Be a centre of excellence for technical education in emerging technologies by exceeding the needs of industry and society.
- Be an institute with world class research facilities.
- Be an institute nurturing talent and enhancing competency of students to transform them as all- round personalities respecting moral and ethical values.

VISION AND MISSION OF THE DEPARTMENT

To excel in education, innovation and research in Artificial Intelligence and Data Science to fulfill industrial demands and societal expectations.

Mission 1: To educate future engineers with solid fundamentals, continually improving teaching methods using modern tools.

Mission 2: To collaborate with industry and offer top-notch facilities in a conductive learning environment.

Mission 3: To foster skilled engineers and ethical innovation in AI and Data Science for global recognition and impactful research.

Mission 4: To tackle the societal challenge of producing capable professionals by instilling employability skills and human values.

PROGRAM EDUCATIONAL OBJECTIVES (PEOS)

PEO 1: Compete on a global scale for a professional career in Artificial Intelligence and Data Science.

PEO 2: Provide industry-specific solutions for the society with effective communication and ethics.

PEO 3: Hone their professional skills through research and lifelong learning initiatives.

PROGRAM OUTCOMES

Engineering students will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSOs)

- **PSO 1:** Capable of working on data-related methodologies and providing industry-focussed solutions.
- **PSO2:** Capable of analysing and providing a solution to a given real-world problem by designing an effective program.

ABSTRACT

This project implements a functional search engine system that indexes documents and web pages to provide efficient retrieval of information based on user queries. The search engine is designed to handle large datasets and uses advanced Java programming concepts for its implementation. Features include indexing files, crawling web pages using the Jsoup library, displaying results via a graphical user interface (GUI) built with Swing, and ranking results based on their relevance to the query. The project aims to demonstrate the fundamentals of search engine architecture while providing a platform for future improvements.

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

1.1 INTRODUCTION

Search engines are at the core of modern information retrieval systems, enabling users to access relevant data quickly and efficiently. This project simulates a basic search engine capable of indexing local files and web pages and performing searches on the indexed content. It uses Java to demonstrate the functionality of search engines, combining concepts of data structures, file handling, and web crawling. The motivation behind the project is to understand how search engines like Google and Bing work and to build a simplified version for academic purposes.

1.2 OBJECTIVE

The primary objectives of this project are:

- 1. To develop a system that indexes documents and web pages effectively.
- 2. To enable keyword-based search functionality with ranked results.
- 3. To implement a user-friendly GUI for performing searches.
- 4. To optimize the search process for large datasets.
- 5. To demonstrate the integration of file management and web crawling in Java.

CHAPTER 2
PROJECT METHODOLOGY

2.1 PROPOSED WORK

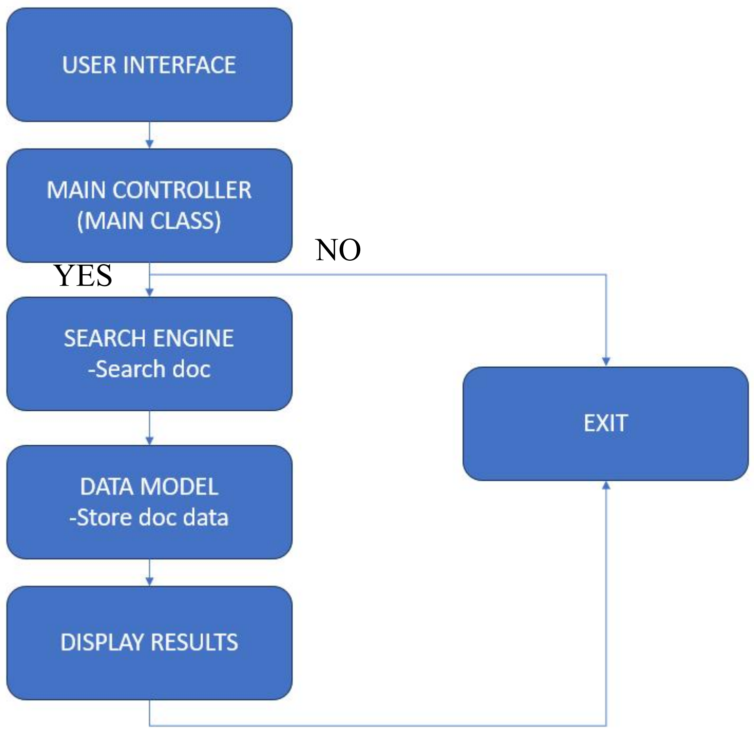
The project is divided into three main phases:

1. **Indexing Phase:** Local files and web pages are processed to extract and store meaningful content in an inverted index. Each term in the index is associated with the files/web pages where it occurs and its frequency.

2. **Search and Ranking Phase:** User queries are matched against the index to identify relevant documents. Results are ranked based on term frequency (TF) to prioritize the most relevant files/web pages.

3. **GUI Phase:** A graphical user interface built with Java Swing allows users to interact with the system by indexing files/web pages and searching for keywords.

2.2 BLOCK DIAGRAM


USER INTERFACE
MAIN CONTROLLER (MAIN CLASS)
NO
YES
SEARCH ENGINE -Search doc
EXIT
DATA MODEL -Store doc data
DISPLAY RESULTS

CHAPTER 3
JAVA PROGRAMMING CONCEPTS

3.1 FILE HANDLING

This involves reading and writing data to files. In the program, it is used to read the contents of text files selected by the user. The content is then processed and indexed so that the program can search through it later. Java classes like FileReader, BufferedReader, and File are typically used for file handling operations.

3.2 COLLECTIONS FRAMEWORK

Java's Collections Framework is used to handle and manipulate groups of data efficiently. In this program, HashMap and HashSet are part of the framework:

- HashMap is used to store the inverted index, which maps words to the documents or URLs where they appear, along with the word frequencies.
- HashSet is used to track visited URLs to avoid redundant crawling.

3.3 SWING

Swing is a Java library for creating graphical user interfaces (GUIs). The program uses Swing components like JFrame (main application window), JTextField (for input), JTextArea (to display results), and JButton (to trigger actions like indexing and searching). The GUI allows users to interact with the application visually instead of using the command line.

3.4 MULTITHREADING

While it isn't explicitly implemented in the code above, multithreading is often used in web crawlers to fetch data from multiple URLs simultaneously. This improves performance by allowing the program to handle multiple tasks concurrently, such as downloading web pages while indexing others.

3.5 EXCEPTION HANDLING

This concept ensures that the program can handle runtime errors gracefully without crashing. For example, when attempting to read a file that doesn’t exist or when there’s a network issue while crawling web pages, the program uses try-catch blocks to manage these errors and provide meaningful messages to the user.

CHAPTER 4
MODULE DESCRIPTION

4.1 INDEXING MODULE

The Indexing Module is responsible for processing the text content from files and web pages to build an inverted index, which maps words to their occurrences across documents or URLs. This module ensures that data is organized efficiently for quick retrieval, using structures like HashMap to store word frequencies and document associations.

4.2 SEARCH MODULE

The Search Module handles user queries by searching the indexed data for matching terms. It retrieves results from the index, sorts them based on frequency or relevance, and formats the output for display. This module is crucial for ensuring accurate and fast retrieval of information.

4.3 GUI MODULE

The GUI Module provides a user-friendly interface for interacting with the application. Built using Java Swing, it includes components like text fields for input, buttons for triggering actions, and text areas for displaying results. This module bridges the technical backend with an intuitive frontend, allowing users to perform tasks like indexing and searching without dealing with command-line operations.

4.4 PERFORMANCE OPTIMIZATION

The Performance Module ensures the application runs efficiently, particularly when handling large datasets or multiple web pages. Techniques like multithreading (if implemented) allow simultaneous processing, such as crawling multiple web pages concurrently, thereby optimizing resource utilization and response times.

4.5 FILE HANDLING

The File Handling Module manages reading and writing operations with local files. It reads content from text files selected by the user, processes it for indexing, and ensures error handling to gracefully manage issues like missing files or unsupported formats. This module is essential for integrating local data into the search engine system.

CHAPTER 5
CONCLUSION

This project successfully demonstrates the fundamental workings of a search engine using Java. Key achievements include:

- Indexing both files and web pages.
- Building a user-friendly GUI for seamless interaction.
- Optimizing performance for efficient data handling.
- Providing a foundation for implementing more advanced features like semantic search or machine learning-based ranking.

Future work could involve integrating natural language processing for better query understanding, enhancing the ranking algorithm, and adding support for more file types and languages.

REFERENCES:

- Java SE Documentation: https://docs.oracle.com/javase/
- Jsoup Library: https://jsoup.org/
- Java Swing Tutorials: https://docs.oracle.com/javase/tutorial/uiswing/

APPENDICES

APPENDIX A – SOURCE CODE

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.io.*;
import java.net.URI;
import java.util.*;
import org.jsoup.Jsoup;
import org.jsoup.nodes.Document;
import org.jsoup.select.Elements;

public class SearchEngineApp extends JFrame {

    private final JTextField searchField;
    private final JTextArea resultArea;
    private final Map<String, Map<String, Integer>> index;
    private final Set<String> visitedURLs;

    public SearchEngineApp() {
        index = new HashMap<>();
        visitedURLs = new HashSet<>();
        setTitle("Search Engine");
        setSize(1000, 700);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new BorderLayout());
```

```
getContentPane().setBackground(new Color(245, 245, 245));
Font customFont = new Font("Arial", Font.PLAIN, 14);
UIManager.put("Label.font", customFont);
JPanel topPanel = new JPanel();
topPanel.setLayout(new FlowLayout(FlowLayout.LEFT));
topPanel.setBackground(new Color(0, 123, 255));
JButton addFilesButton = new JButton("Index Files & Web Pages");
addFilesButton.setBackground(new Color(70, 130, 180));
addFilesButton.setForeground(Color.WHITE);
addFilesButton.addActionListener(e -> indexFilesAndWebPages());

JLabel searchLabel = new JLabel("Search:");
searchLabel.setForeground(Color.WHITE);

searchField = new JTextField(25);
searchField.setFont(new Font("Arial", Font.PLAIN, 14));

JButton searchButton = new JButton("Search");
searchButton.setBackground(new Color(255, 140, 0));
searchButton.setForeground(Color.WHITE);
searchButton.addActionListener(e -> performSearch());
topPanel.add(addFilesButton);
topPanel.add(searchLabel);
topPanel.add(searchField);
topPanel.add(searchButton);
resultArea = new JTextArea();
resultArea.setEditable(false);
resultArea.setLineWrap(true);
resultArea.setBackground(new Color(255, 255, 255));
```

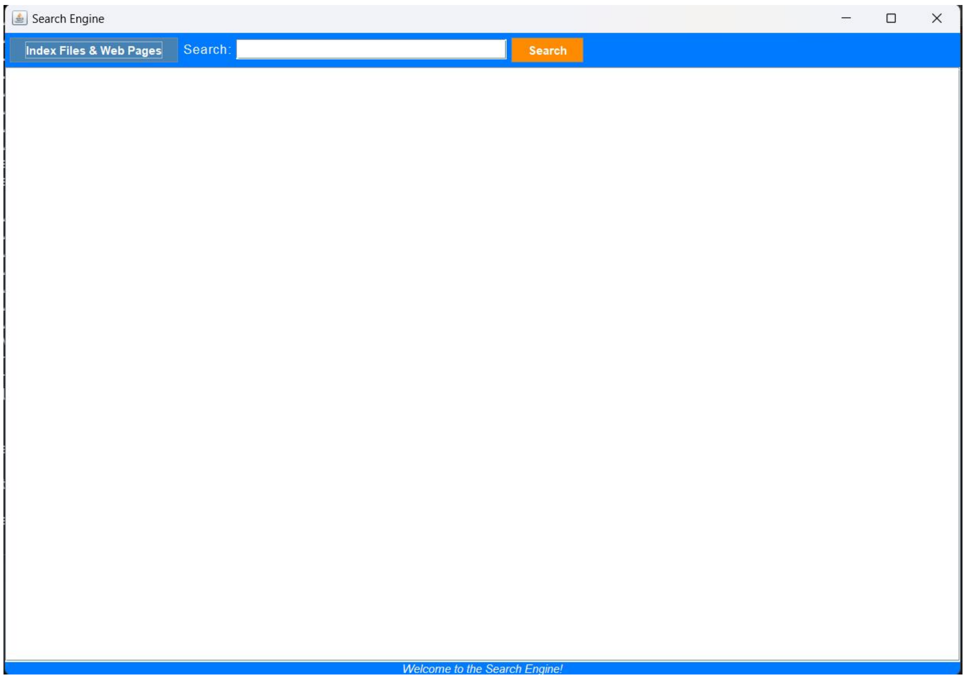
```
        resultArea.setFont(new Font("Monospaced", Font.PLAIN, 14));
        resultArea.setForeground(new Color(50, 50, 50));
        JScrollPane resultScrollPane = new JScrollPane(resultArea);
        add(topPanel, BorderLayout.NORTH);
        add(resultScrollPane, BorderLayout.CENTER);
        JLabel statusBar = new JLabel("Welcome to the Search Engine!",
JLabel.CENTER);
        statusBar.setFont(new Font("Arial", Font.ITALIC, 12));
        statusBar.setBackground(new Color(0, 123, 255));
        statusBar.setForeground(Color.WHITE);
        statusBar.setOpaque(true);
        add(statusBar, BorderLayout.SOUTH);
        setVisible(true);
    }

    private void indexFilesAndWebPages() {
        JFileChooser fileChooser = new JFileChooser();
        fileChooser.setMultiSelectionEnabled(true);
        int returnValue = fileChooser.showOpenDialog(this);
        if (returnValue == JFileChooser.APPROVE_OPTION) {
            File[] selectedFiles = fileChooser.getSelectedFiles();
            for (File file : selectedFiles) {
                try {
                    indexFile(file);
                } catch (IOException e) {
                    showError("Error indexing file: " + file.getName());
                }
            }
        }
```

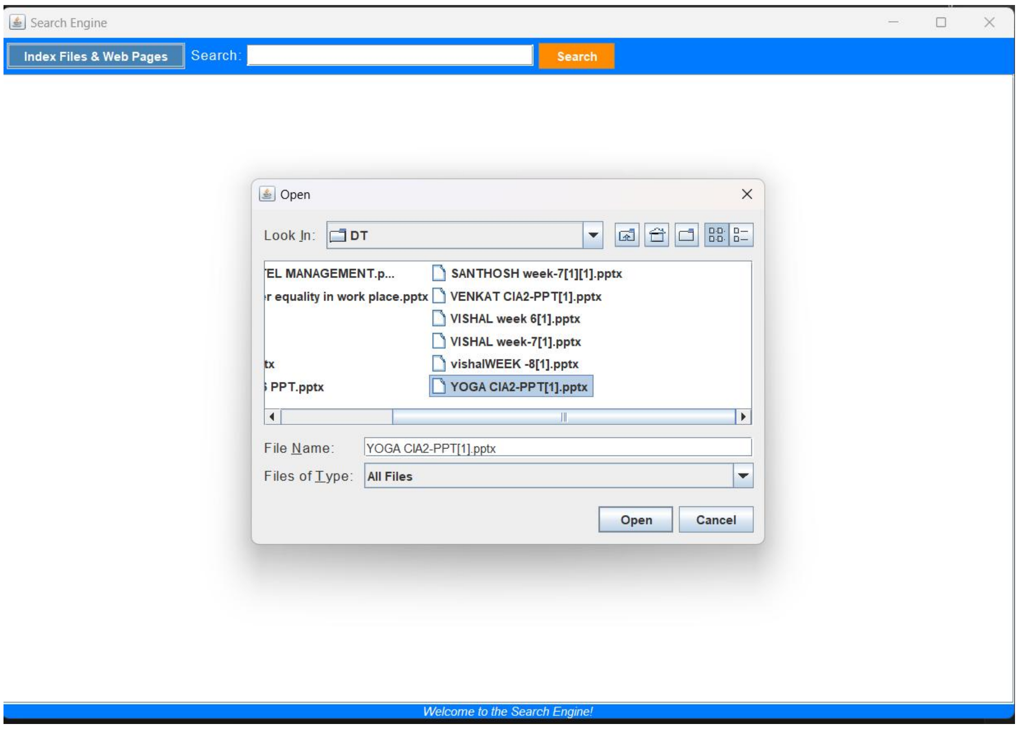
```
            String url = JOptionPane.showInputDialog(this, "Enter URL to index:");
            if (url != null && !url.isEmpty()) {
                if (!url.startsWith("http://") && !url.startsWith("https://")) {
                    url = "http://" + url; // Add the HTTP protocol if missing
                }
                try {
                    crawlAndIndexWebPage(url);
                } catch (IOException e) {
                    showError("Error indexing web page: " + url);
                }
            }

            JOptionPane.showMessageDialog(this, "Indexing completed!");
        }

        private void crawlAndIndexWebPage(String url) throws IOException {
            if (visitedURLs.contains(url)) return;

            visitedURLs.add(url);
            Document                                doc                              =
    Jsoup.connect(url).userAgent("Mozilla/5.0").timeout(5000).get();
            Elements body = doc.body().select("p"); // Select all paragraph texts
            StringBuilder content = new StringBuilder();
            body.forEach(element -> content.append(element.text()).append(" "));
            indexDocument(url, content.toString());
            Elements links = doc.select("a[href]"); // Select all links in the web page
            for (org.jsoup.nodes.Element link : links) {
                String linkURL = link.attr("href");
                if (!linkURL.startsWith("http")) {
```

```
                linkURL = doc.baseUri() + linkURL;
            }

            if (!visitedURLs.contains(linkURL)) {
                crawlAndIndexWebPage(linkURL);
            }
        }
    }

    private void indexFile(File file) throws IOException {
        if (file == null || !file.exists()) {
            throw new FileNotFoundException("File not found: " + file);
        }

        try (BufferedReader br = new BufferedReader(new FileReader(file))) {
            StringBuilder content = new StringBuilder();
            String line;
            while ((line = br.readLine()) != null) {
                content.append(line).append(" ");
            }
            indexDocument(file.getName(), content.toString());
            resultArea.setText("File indexed: " + file.getName());
        }
    }

    private void indexDocument(String docId, String content) {
        String[] words = content.split("\\W+");
        for (String word : words) {
            word = word.toLowerCase();
```

```
                index.putIfAbsent(word, new HashMap<>());
                Map<String, Integer> postings = index.get(word);
                postings.put(docId, postings.getOrDefault(docId, 0) + 1);
            }
        }

        private void performSearch() {
            String query = searchField.getText().toLowerCase().trim();
            if (query.isEmpty()) {
                resultArea.setText("Please enter a search term.");
                return;
            }
          Map<String,Integer>results=index.getOrDefault(query,Collections.empty
Map());
            if (results.isEmpty()) {
                resultArea.setText("No results found for: " + query);
                return;
            }
            ArrayList<Map.Entry<String,     Integer>>     sortedResults     =     new
ArrayList<>(results.entrySet());
            sortedResults.sort((e1, e2) -> e2.getValue().compareTo(e1.getValue()));

            StringBuilder sb = new StringBuilder();
            sb.append("Results for '").append(query).append("':\n\n");
            for (Map.Entry<String, Integer> entry : sortedResults) {
                sb.append("[CLICK                    TO                    OPEN]
").append(entry.getKey()).append("(").append(entry.getValue()).append("
occurrences)\n");
            }
            resultArea.setText(sb.toString());
            addClickListenersToResults(sortedResults);
        }

        private    void    addClickListenersToResults(ArrayList<Map.Entry<String,
Integer>> sortedResults) {
            resultArea.addMouseListener(new MouseAdapter() {
                @Override
                public void mousePressed(MouseEvent e) {
                    String resultText = resultArea.getText();
                    Point clickPoint = e.getPoint();
                    int position = resultArea.viewToModel(clickPoint);
                    try {
                        String[] lines = resultText.split("\n");
                        for (String line : lines) {
                            if (line.contains("[CLICK TO OPEN]")) {
                          String filename = line.substring(15, line.indexOf(" ("));
                                File file = new File(filename);
                                if (file.exists()) {
                                    Desktop.getDesktop().open(file);
                                } else if (filename.startsWith("http")) {
                            Desktop.getDesktop().browse(new URI(filename));
                                }
                            }
                        }
                    } catch (IOException | URISyntaxException ex) {
                        showError("Error opening the file or URL.");
                    }
            }
        });
    }

    private void showError(String message) {
        JOptionPane.showMessageDialog(this,       message,
"Error", JOptionPane.ERROR_MESSAGE);
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            new SearchEngineApp(); // Run the app
        });
    }
}
```

APPENDIX B - SCREENSHOTS

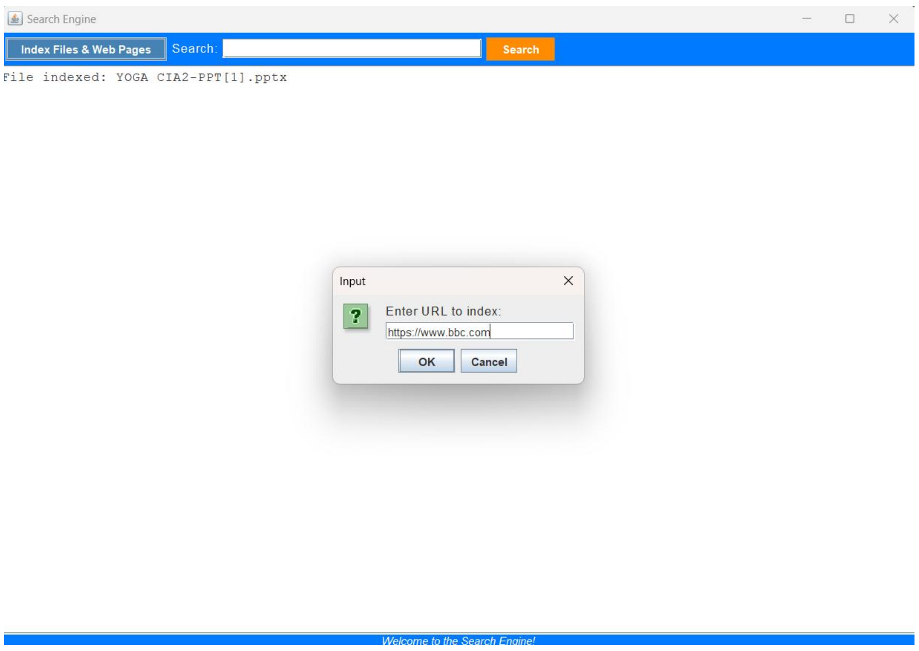




EL MANAGEMENT.p...
r equality in work place.pptx
tx
PPT.pptx
SANTHOSH week-7[1][1].pptx
VENKAT CIA2-PPT[1].pptx
VISHAL week 6[1].pptx
VISHAL week-7[1].pptx
vishalWEEK -8[1].pptx
YOGA CIA2-PPT[1].pptx

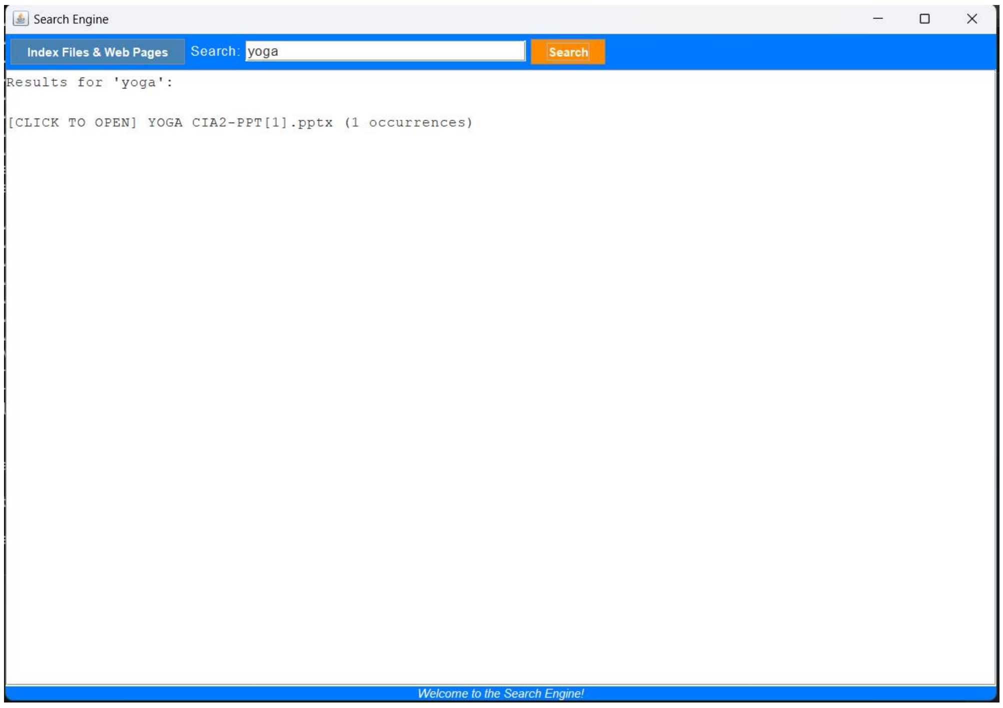
File Name: YOGA CIA2-PPT[1].pptx
Files of Type: All Files

19

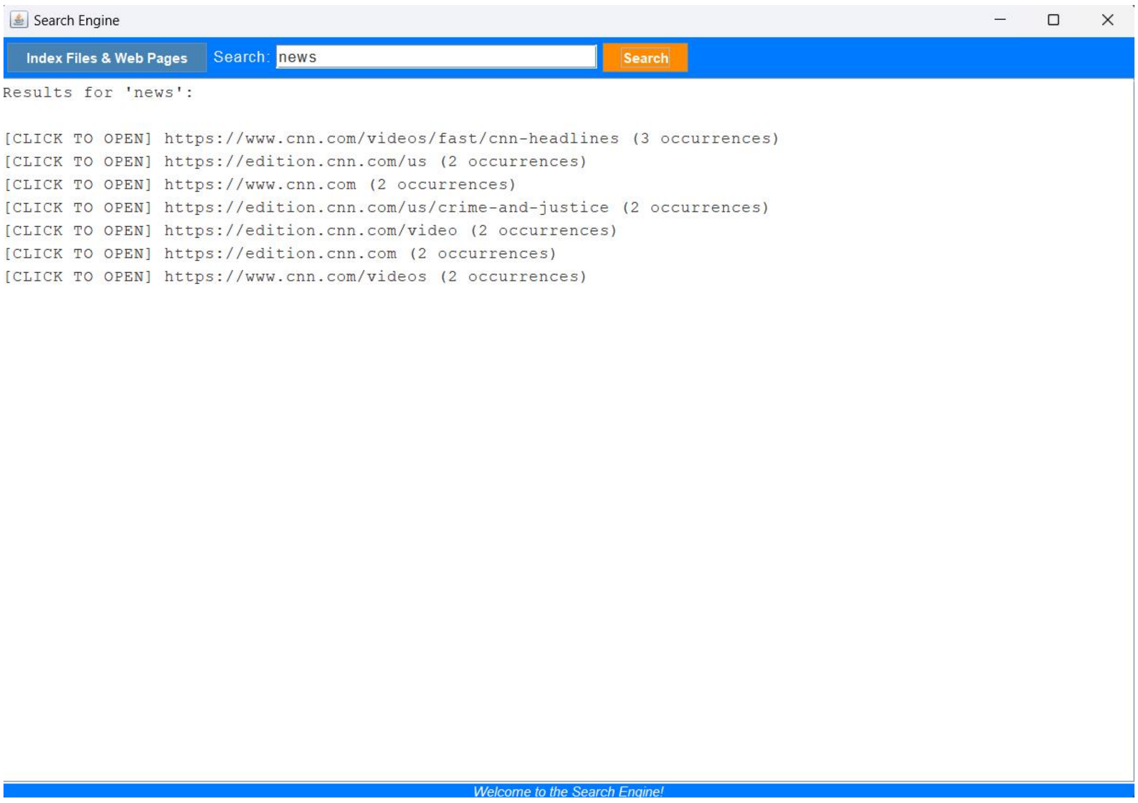


```
File indexed: YOGA CIA2-PPT[1].pptx
```

https://www.bbc.com



```
Results for 'yoga':

[CLICK TO OPEN] YOGA CIA2-PPT[1].pptx (1 occurrences)
```



```
Results for 'news':

[CLICK TO OPEN] https://www.cnn.com/videos/fast/cnn-headlines (3 occurrences)
[CLICK TO OPEN] https://edition.cnn.com/us (2 occurrences)
[CLICK TO OPEN] https://www.cnn.com (2 occurrences)
[CLICK TO OPEN] https://edition.cnn.com/us/crime-and-justice (2 occurrences)
[CLICK TO OPEN] https://edition.cnn.com/video (2 occurrences)
[CLICK TO OPEN] https://edition.cnn.com (2 occurrences)
[CLICK TO OPEN] https://www.cnn.com/videos (2 occurrences)
```