TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| Chapter No. | Title | Page No. |
| 1 | Introduction | 3 |
| 2 | Abstract | 5 |
| 3 | Project Overview | 6 |
| 4 | IR TECHNIQUES USED | 6 |
| 5 | CODE IMPLEMENTATION | 8 |
| 6 | CODE EXPLANATION | 14 |
| 7 | GUI | 15 |
| 8 | CONCLUSION | 19 |

**Introduction**

* Our project is an advanced research paper retrieval system designed to streamline the process of sourcing academic papers across various fields of study, with a particular focus on computer science. The system serves as a centralized platform that employs pivotal

text search algorithms to help researchers, students, and academics quickly find relevant literature. Leveraging a comprehensive indexing strategy, the retrieval system provides a user-friendly interface for querying a vast database of documents, utilizing natural language processing to ensure that users receive the most pertinent results.

* The core of the system is Whoosh, a fast, featureful full-text indexing and searching library implemented in pure Python. Whoosh plays a pivotal role in creating an efficient searchable index and in processing search queries with its powerful scoring algorithms, like BM25F, ensuring that the results are not only relevant but also ranked by their significance and relevance to the user's input. The system smartly handles multi-field queries, allowing users to filter results based on specific document attributes like title, abstract, and content, thereby offering a high degree of search precision.
* The project incorporates Streamlit to create a sleek and responsive front-end application. It translates to a seamless experience where users can enter search queries and receive instant feedback. This interface is straightforward, eliminating the steep learning curve often associated with academic research tools, which means users can dive straight into their research without needing to navigate complex search syntax.

**Abstract**

The Research Paper Retrieval System project presents a sophisticated research paper retrieval system specifically tailored to the expansive domain of computer science. Utilizing the robust capabilities of Whoosh, a pure Python search engine library, the system provides an advanced text indexing and searching framework that efficiently handles large volumes of academic papers. Our system integrates a field-weighted search algorithm, BM25F, to optimize relevance scoring and ensure the high accuracy of search results. The multi-field query capability allows users to refine searches across document titles, abstracts, and content, enhancing the retrieval of pertinent literature.

The user interface is developed using Streamlit, which offers a streamlined and intuitive user experience.

**Project Overview**

* The "Research Paper Retrieval System " project aims to develop a user-friendly literature paper retrieval system that can efficiently find relevant papers from a database given a user-input query. The project encompasses several key components:
* **Natural Language Processing:** At the forefront is the employment of natural language processing (NLP) techniques to understand and interpret user queries. The system harnesses Whoosh's full-text indexing capabilities to parse and match search terms(Multifield parser) with a vast collection of documents, ensuring that relevant papers are easily retrievable.
* **Interactive Web Interface**: Central to the user experience is a Streamlit-based web interface that provides a seamless and straightforward portal for the retrieval system. Users can input their queries, and interact with the search results all within an easy-to-navigate environment.
* **Advanced Feature Set:** Parallel to the visual feature extraction seen in image retrieval systems, this project leverages text-based feature extraction to analyze and understand the content of research papers. The system is equipped to handle complex queries, including semantic search capabilities, which enables users to find papers that are not just keyword matches but are contextually related to the search topic..
* Dynamic Content Acquisition: The system stands out for its dynamic approach to content updating. A combination of web scraping technologies is utilized to continuously ingest new research papers into the database. The project goes a step further by incorporating PDF processing, allowing the extraction of text and metadata from research papers, which is an essential format in scholarly communications

Overall the "Research Paper Retrieval System" project combines state-of-the-art NLP and search ranking algorithms with a focus on user experience to provide robust solution for academic literature retrieval. It stands as a testament to the synergy between advanced computing techniques and practical application, aiming to improve the way researchers access and engage with scholarly content.

**IR Techniques Used**

1. **Text Extraction**: The project uses pdftotext python library to extract text from the pdf file .
2. **Text Analysis and Processing**

Stemming and Lemmatization: Words are reduced to their base or root form, enabling the system to match different forms of a word to the same search query.

Stop-word Removal: Common words that offer little value in search (like "and", "the", "of") are removed to enhance search efficiency.

After preprocessing, only alphabets are taken and converted into tokens

Indexing:

Using woosh indexing function to generate indexes from the tokens of the content , title and abstract.

1. **Indexing:**

Using woosh’s create\_in function for indexing to generate indexes from the tokens of the content , title and abstract.

1. **Ranking Algorithm**:

BM25F: An advanced relevance scoring algorithm that ranks documents based on term frequency, document length, and field-specific weights, considering both the importance of each term and the structure of the document. It considers multiple fields for ranking or for relevance score they are title, abstract and content(extracted tokens from pdf)

1. **Query Parsing**

Syntax Parsing: Queries are parsed to understand the logical structure, identifying phrases, required terms, optional terms, and negations.

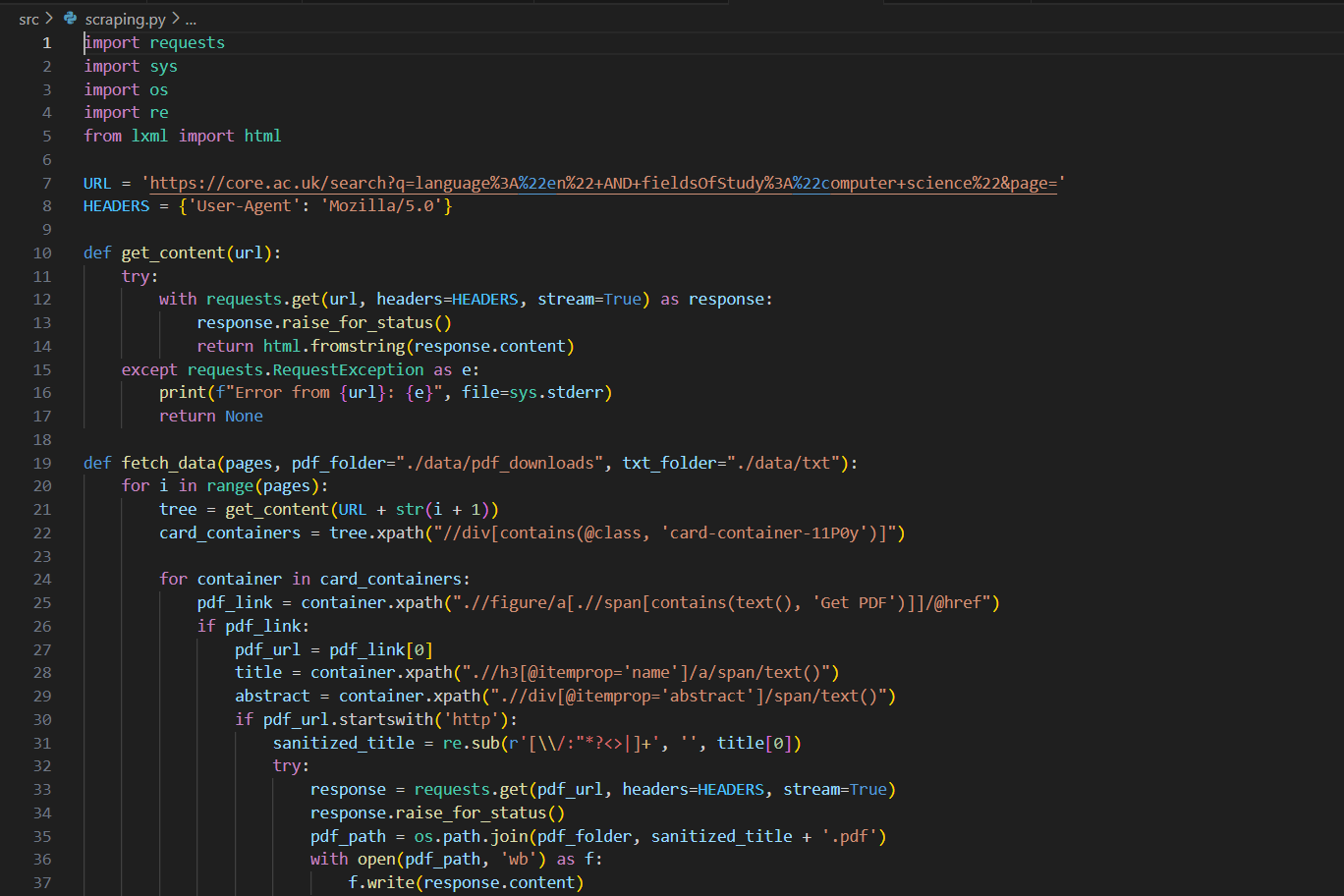
Multifield Queries: Supports queries that span multiple document fields, such as title, abstract, and content, allowing users to specify where to look for information .

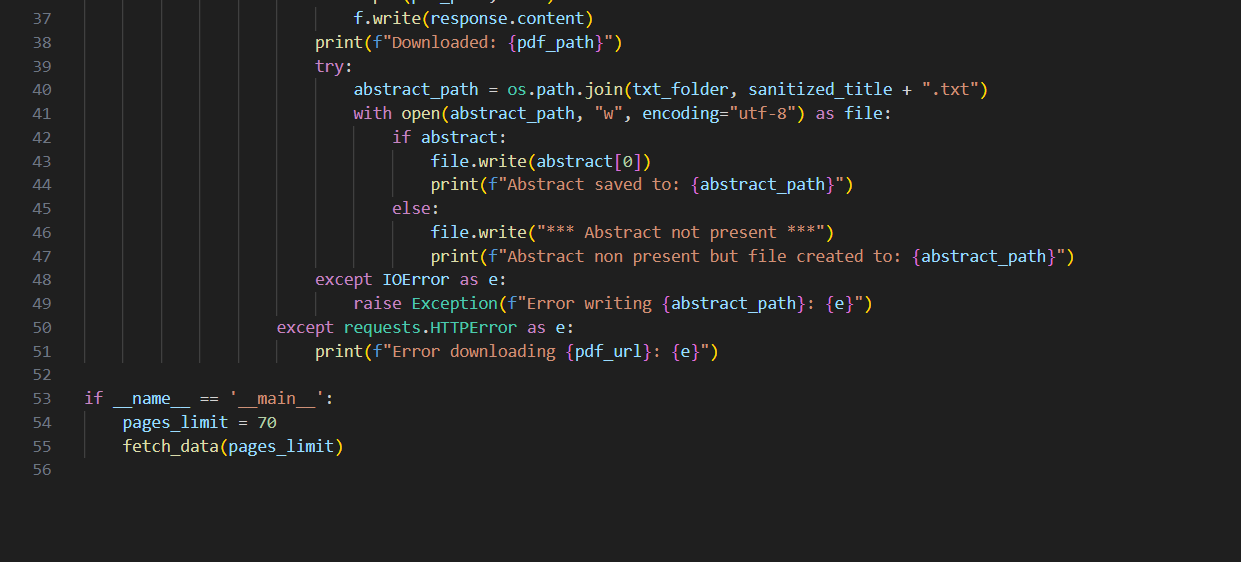
Queries are also pre-processed .

These IR techniques combine to create a powerful system that can understand, process, and retrieve academic papers in a way that aligns with user needs and search behaviors. By leveraging these diverse methodologies, the project aims to deliver an efficient, accurate, and user-friendly literature retrieval experience.

**Code Implementation**

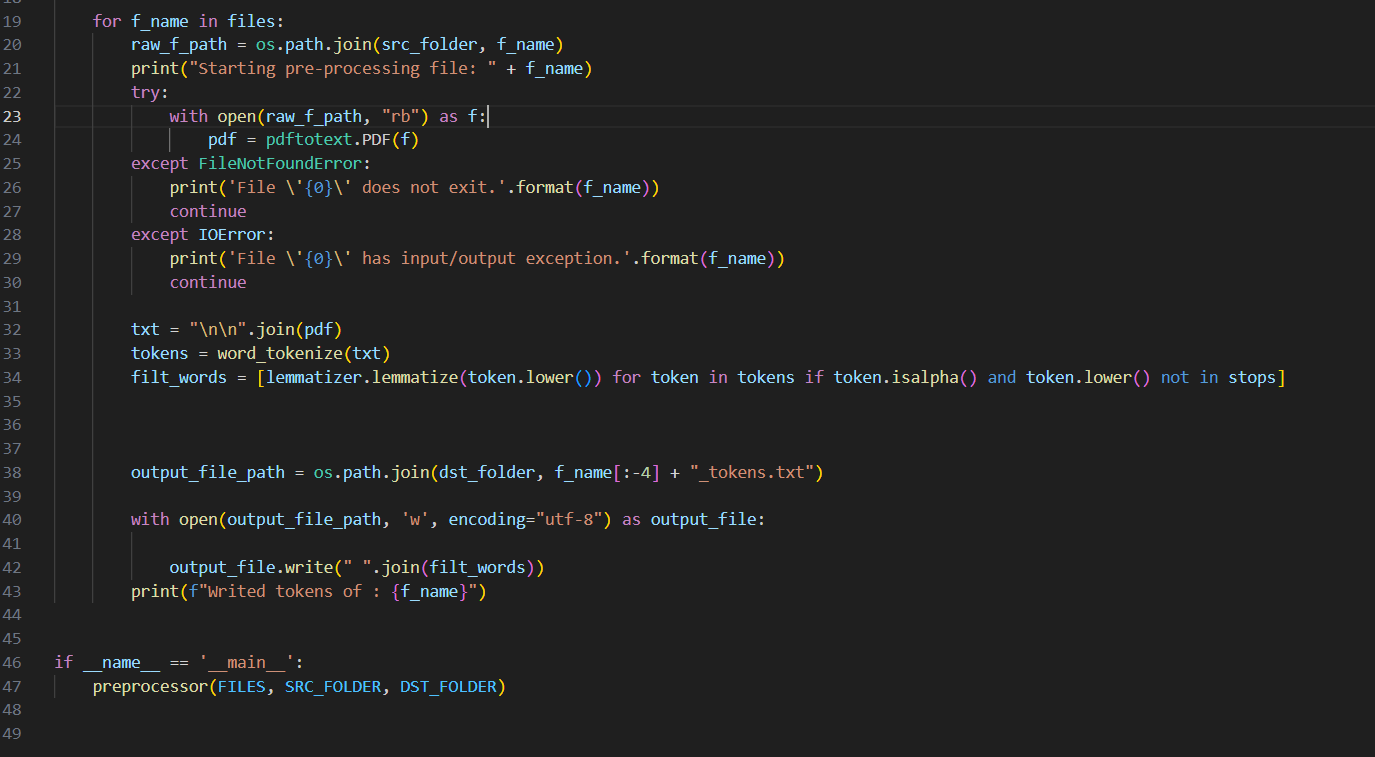
scraping.py



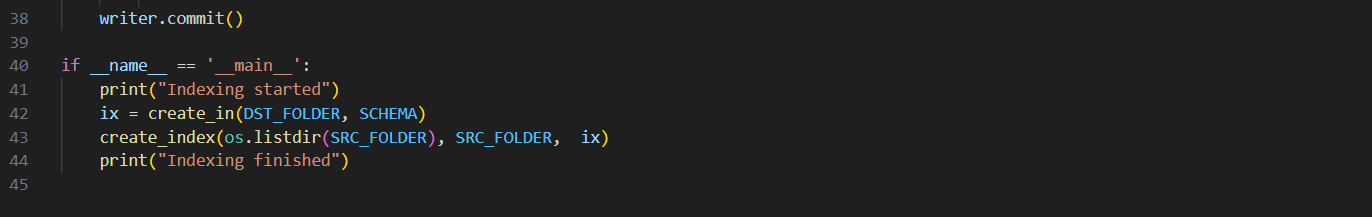
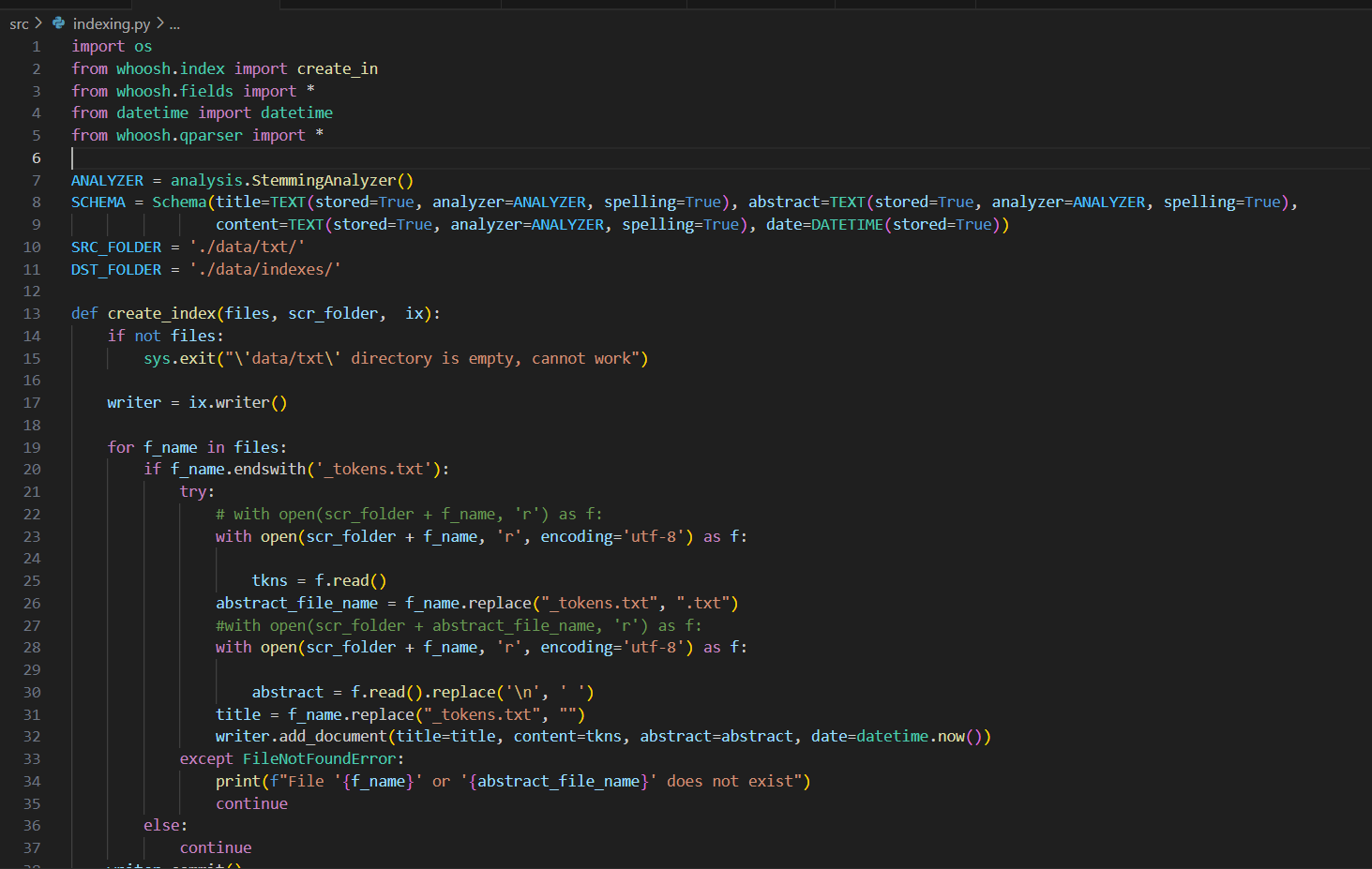


preprocessing.py

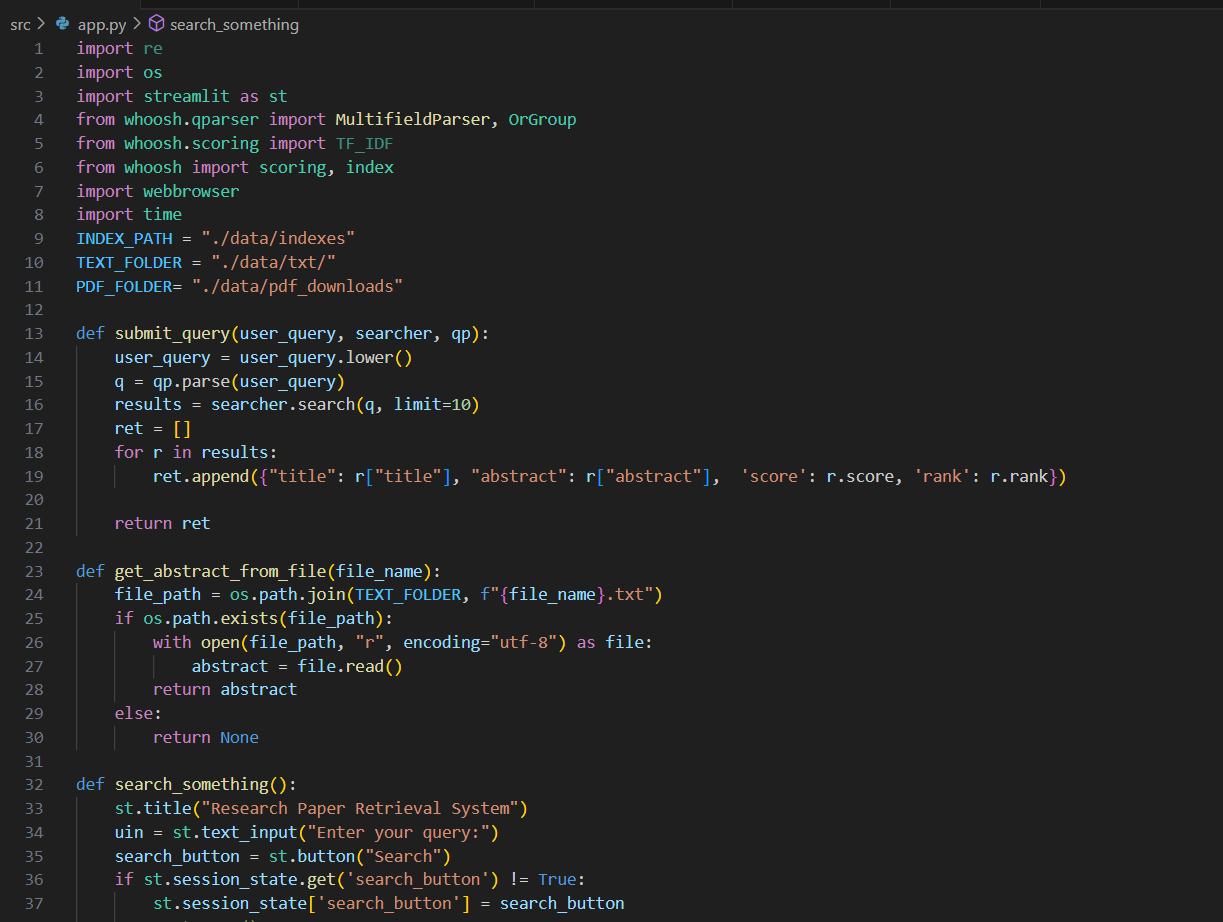


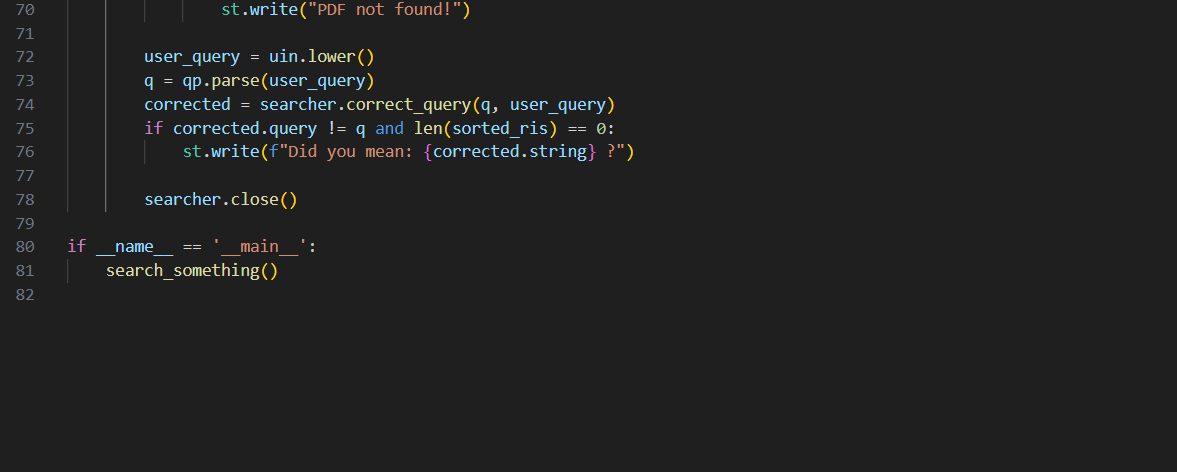
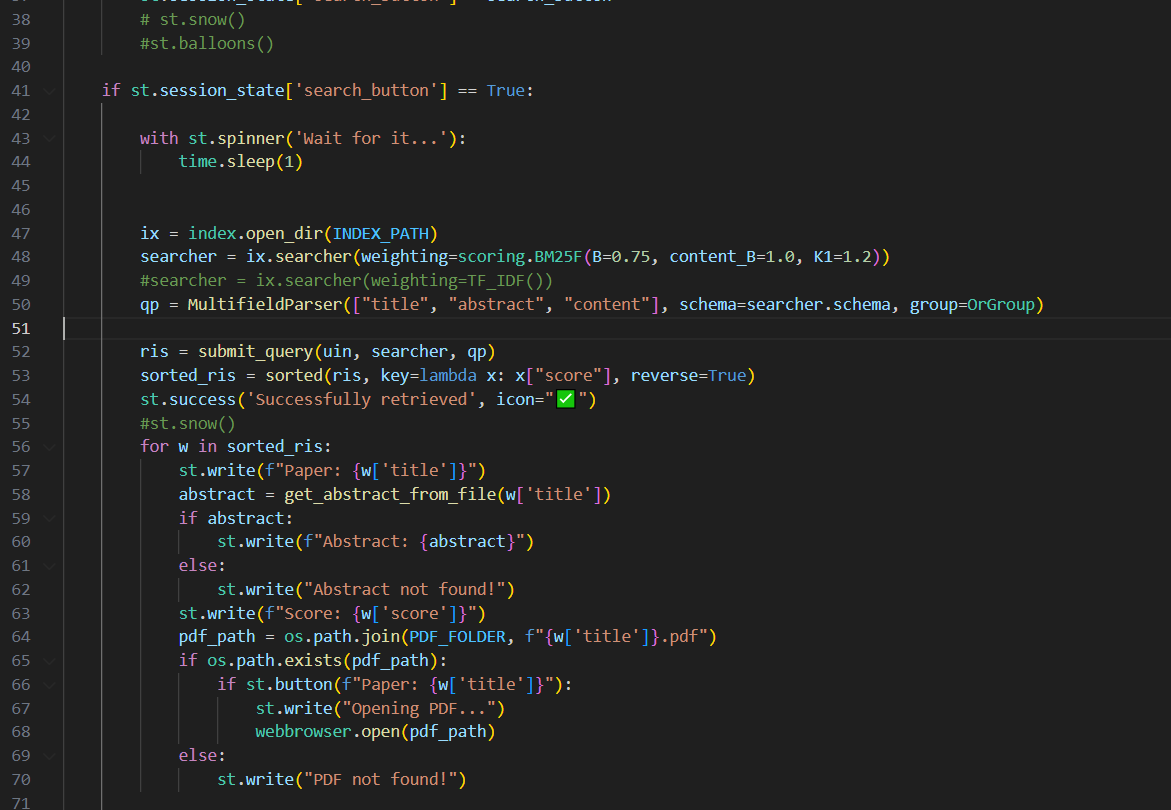


indexing.py



app.py





**CODE Explanation**

1) scraping.py

* This script is used to scrape the core.ac.uk webpage for the computer science related research papers and store the pdf in the data/pdf\_downloads directory

And abstract displayed on the webpage in data/txt directory.The files are stored using title as their file name.

2) preprocessing.py:

* In in this script the pdf files are opened and text is extracted from them using pdftotext library and the extracted text is preprocessed and converted into tokens and stored in the data/txt directory in text file with filename as title\_tokens.

3) indexing.py:

* This script serves to index documents using the Whoosh search engine library. It begins by defining an analyzer to stem and correct spelling in text. The schema is established to define the structure of the index, encompassing fields like title, abstract, content, and date. The create\_index function orchestrates the indexing process, iterating through files in a specified source folder. For each file, it extracts title, abstract, and content based on naming conventions, then adds them to the index. This script streamlines the creation of a search index, facilitating efficient retrieval of relevant documents using Whoosh

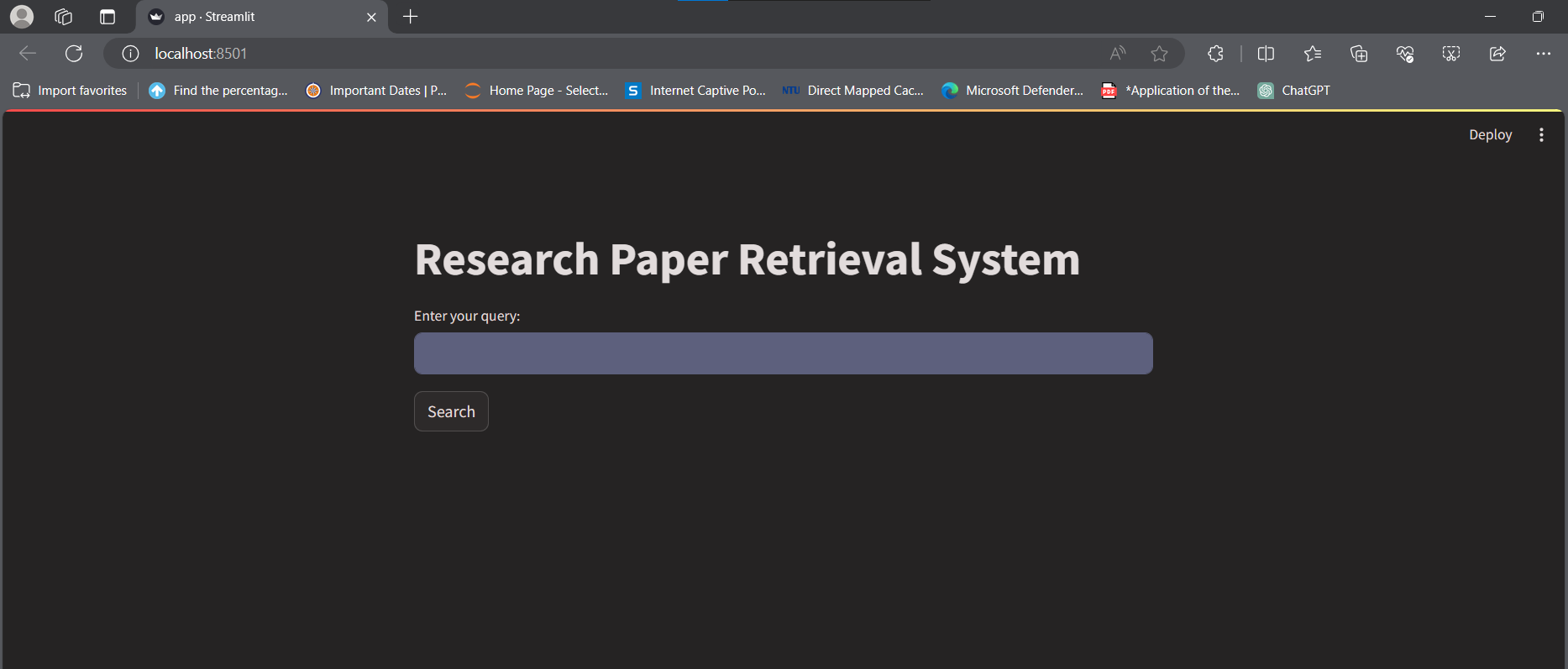
4) app.py:

* This script builds a web-based research paper retrieval system using the Streamlit framework. It leverages the Whoosh library for indexing and searching documents. Upon running, the user is prompted to input a query via a text input field. Upon submitting the query by clicking the search button, the script initiates the search process. It opens the Whoosh index, parses the user's query, and searches for relevant documents. It uses BM25F algorithm for ranking and Multifield parser to parse. The retrieved papers are then displayed in descending order of relevance, showing the paper title, abstract and relevance score. For each paper, if a corresponding PDF file is found, a button is provided to open the PDF. Additionally, the script suggests query corrections if no results are found and displays a success message upon successful retrieval. Overall, this system provides a user-friendly interface for searching and accessing research papers, enhancing the efficiency of literature review and information retrieval tasks.

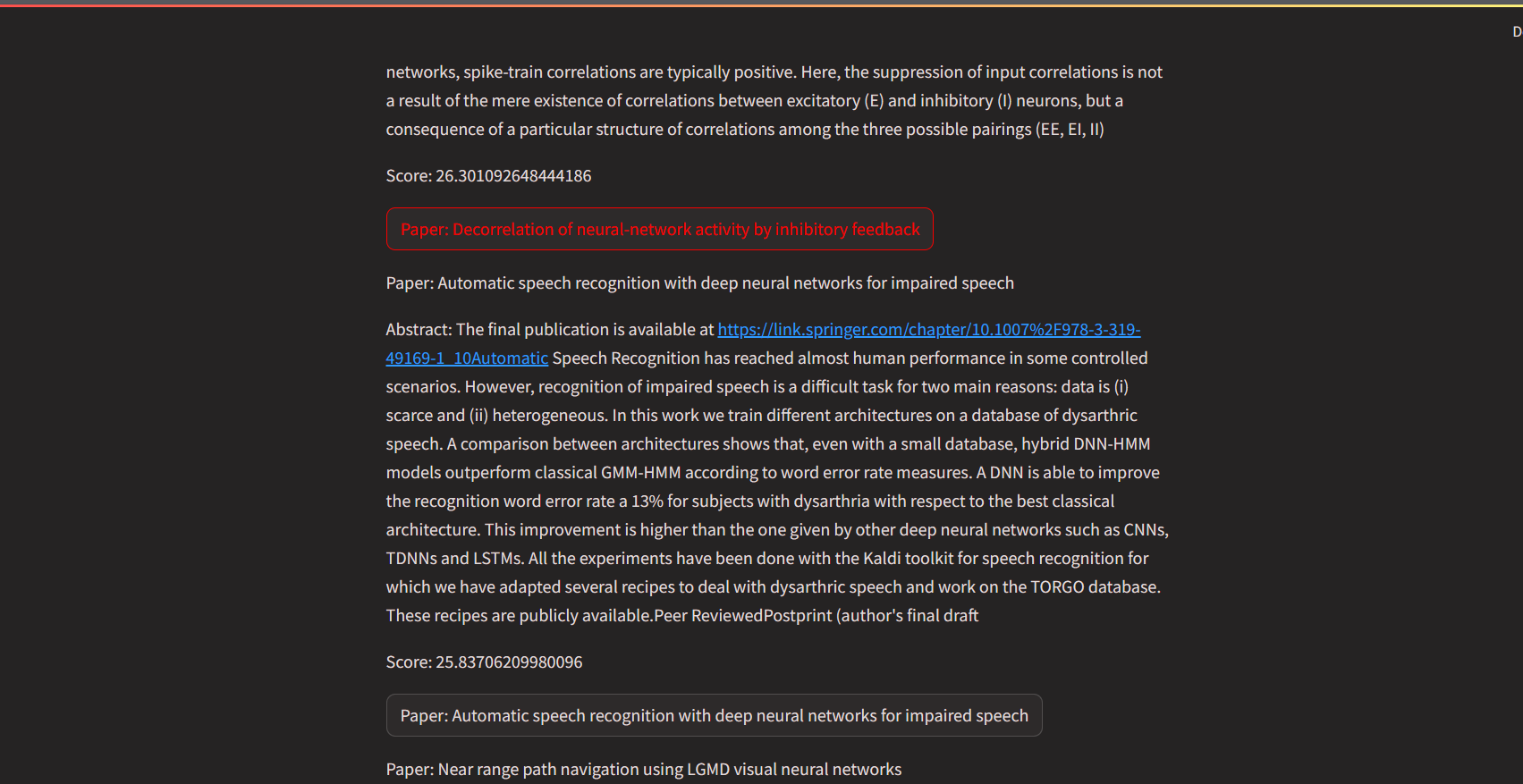
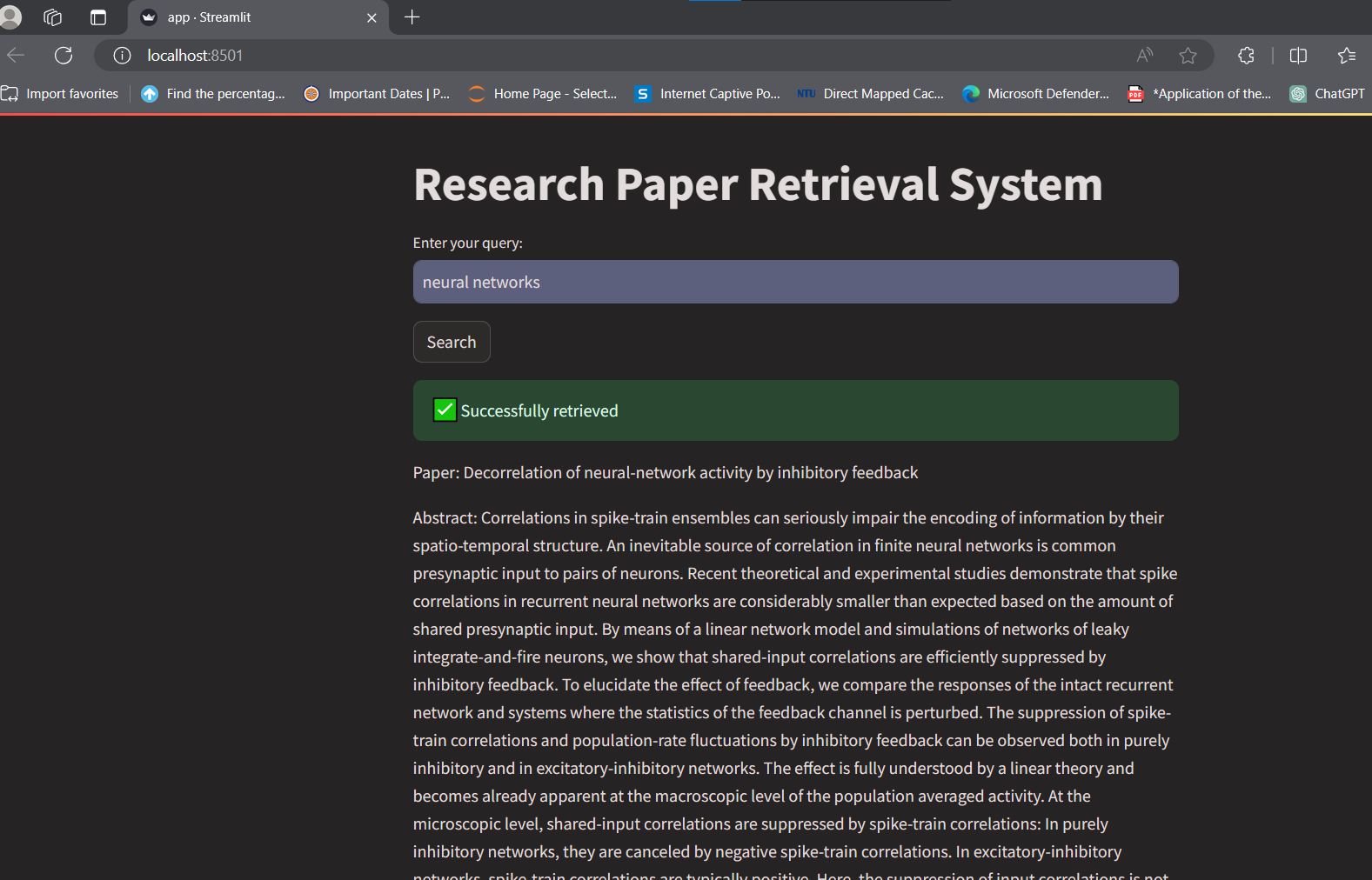
**GUI**

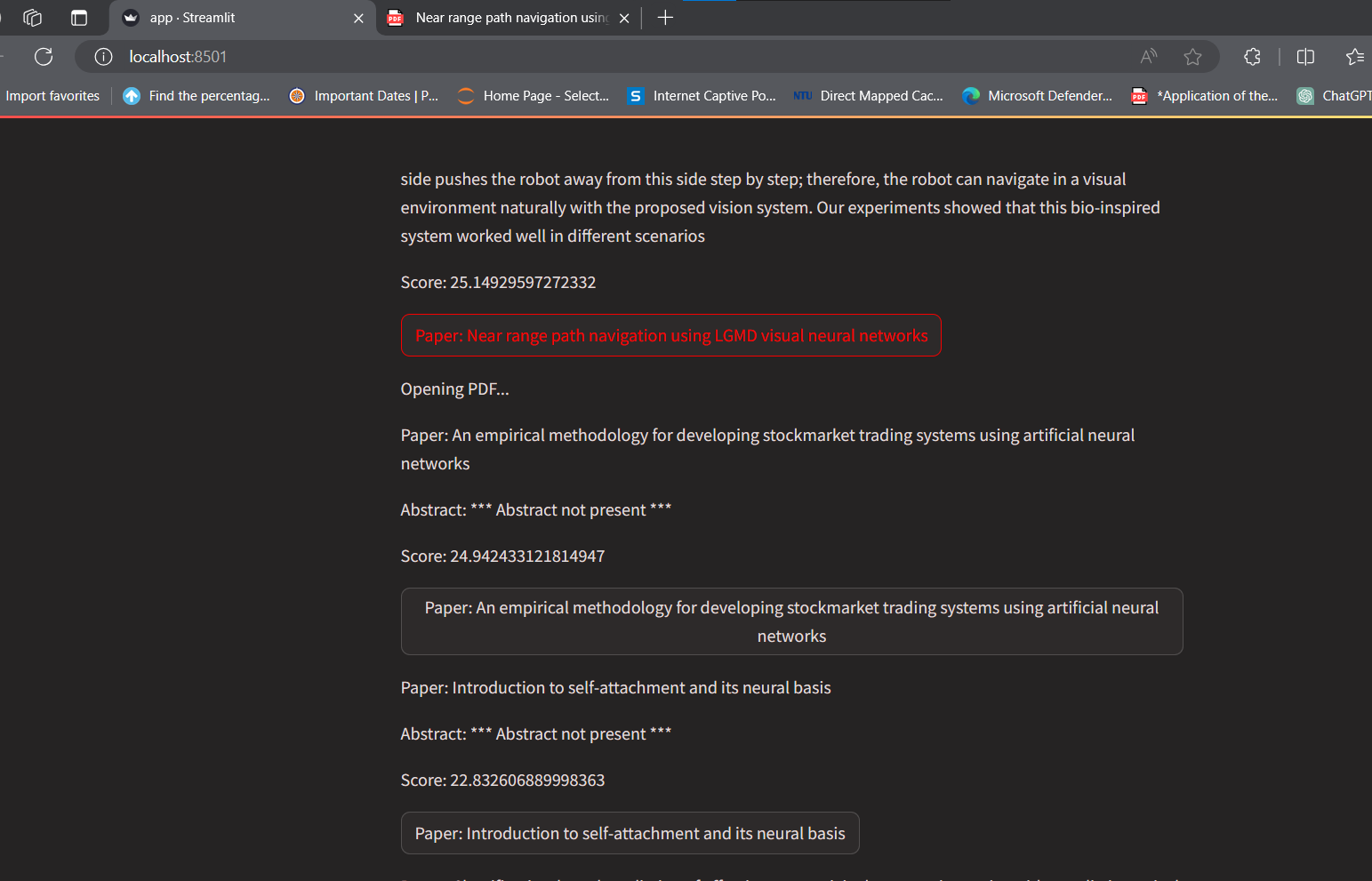
**1.** **Title and Introduction:**

* The GUI starts with a title "Information Retrieval System" with a text input box for query and a search.



**2. Result:**when the search button is clicked the results (retrieved papers) are displayed with title ,abstract and button which on click opens the paper.





**Conclusion**

In conclusion, the Research Paper Retrieval System represents a significant advancement in the domain of academic literature search. By harnessing the power of state-of-the-art information retrieval techniques and a user-centered design, the system offers an unparalleled tool for researchers, students, and academics to access a wealth of scientific knowledge. The integration of full-text search, intelligent ranking algorithms like BM25F, and sophisticated natural language processing ensures that users are presented with the most relevant documents that match their search intent.The system's robust architecture, which incorporates dynamic content updating through web scraping and PDF text extraction, guarantees an ever-evolving and up-to-date repository of research papers. Coupled with an intuitive web interface, the project simplifies the typically complex task of academic research, enabling efficient and effective scholarly inquiries.