

**Department of Computer Science**

**College of Engineering and Computing Science**

**Information Retrieval**

**CSCI 426**

**Fall 2018**

**Sports News Search Engine**

Dr. Houwei Cao

|  |  |
| --- | --- |
| Hasol Im  [him@nyit.edu](mailto:him@nyit.edu)  1073286 | Mansi Parikh  [mparik15@nyit.edu](mailto:mparik15@nyit.edu)  1078420 |
| Myint Hein  [mhein@nyit.edu](mailto:mhein@nyit.edu)  1075461 | Qijun Liang  [qliang05@nyit.edu](mailto:qliang05@nyit.edu)  1073366 |

December 4, 2018

[**Abstract**](#_538kmmod6kyb) **2**

[**Background**](#_bo2v2vke88kj) **2**

[**Related Work**](#_3xjg8pbmercv) **2**

[**Data Collection**](#_xjpa9m1sasa9) **2**

[**Methods**](#_lu1lcrrnzkwj) **3**

[Data Collection](#_ikcpw08aelcj) 3

[Tokenization and Stopword Removal](#_yd544c7jez67) 4

[TF-IDF weighting](#_ngad1rs1tzkk) 4

[Query processing](#_he8jjoc964wx) 5

[Web Interface](#_g1cy85q1e0o) 6

[**Result**](#_r8k8q13678f1) **6**

[**Conclusion**](#_sxzvj85vtssr) **8**

[**Contribution**](#_atir10cnajx8) **8**

[**Reference**](#_xmci3qlr3re6) **8**

# Abstract

The objective of this project is to show a traditional IR model retrieves data and how it computes the query to return search results. We created a web based program which can be used to search for online articles using python.We used python because it gave us more advantage over other programming languages. Our program is capable of crawling articles from other article websites. With the articles stored, our algorithm allows the user to enter a query. The query will be compared with the corpus and list relevant article based on cosine similarity.

# Background

There are a lot of article publication websites out there and readers are having trouble picking a particular publisher to subscribed to. Therefore, our project’s goal is to round up all the popular publisher websites and collect the articles. This will allow users to use our website and be able to view articles from multiple websites without having to access them and look for articles.

# Related Work

Some of the related work would be the search functions in the article websites. Similarities would be that both the web publisher and our program searches the article body for the query keyword and display based on the cosine similarity.

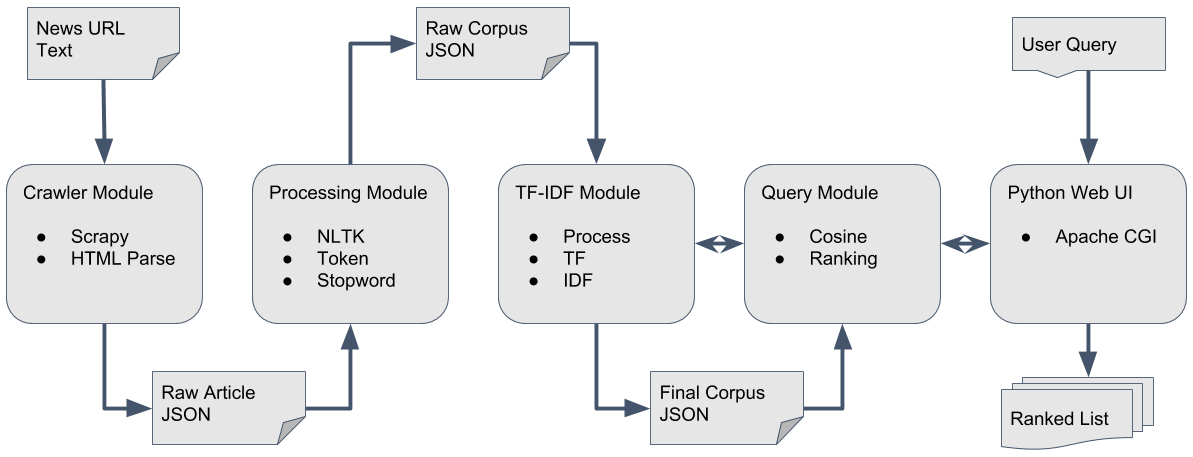
# Data Collection

Instead of using dummy data, we decided to use real articles that are from publication websites. Data was collected from various news sites: New York Post, NBC new, ABC news, Fox news, NPR news. Scrapy was used to crawl data from these websites. There are total of 252 articles in document corpus.

Following list shows base urls used for the web crawling.

* https://nypost.com/sports/
* https://profootballtalk.nbcsports.com
* https://abcnews.go.com/Sports
* https://www.foxnews.com/sports
* https://www.npr.org/sections/sports/

# Methods



The program consists of five modules.

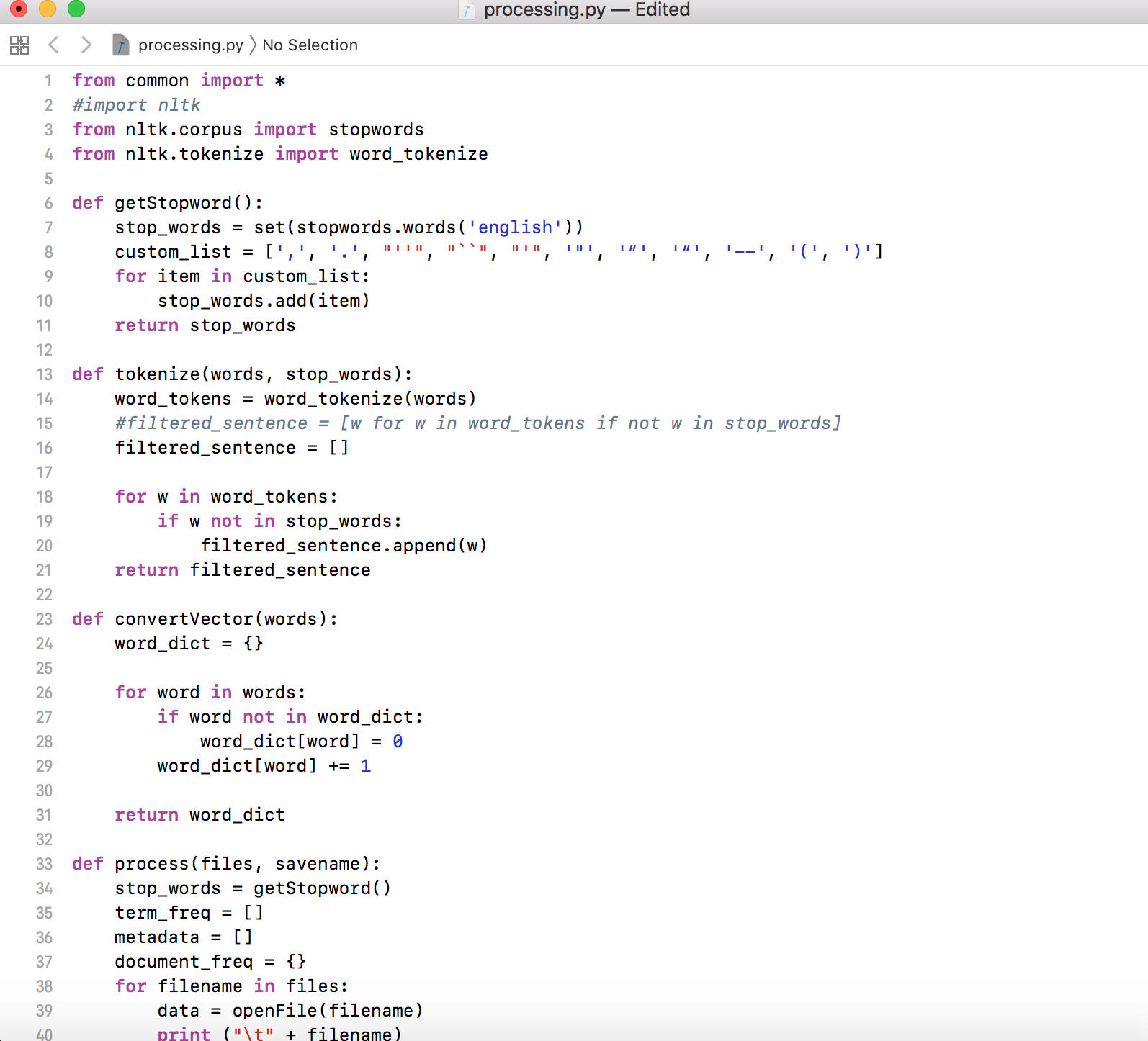
## Data Collection

First we needed to collect data in order to create the IR model. We decided to use real data. We collected the data from online news articles using web crawler.

At this point, the news article is parsed to find the relevant HTML tags. CSS-based parsing was used to extract information of Article, Title, Author, and Date. The result is written into JSON file.

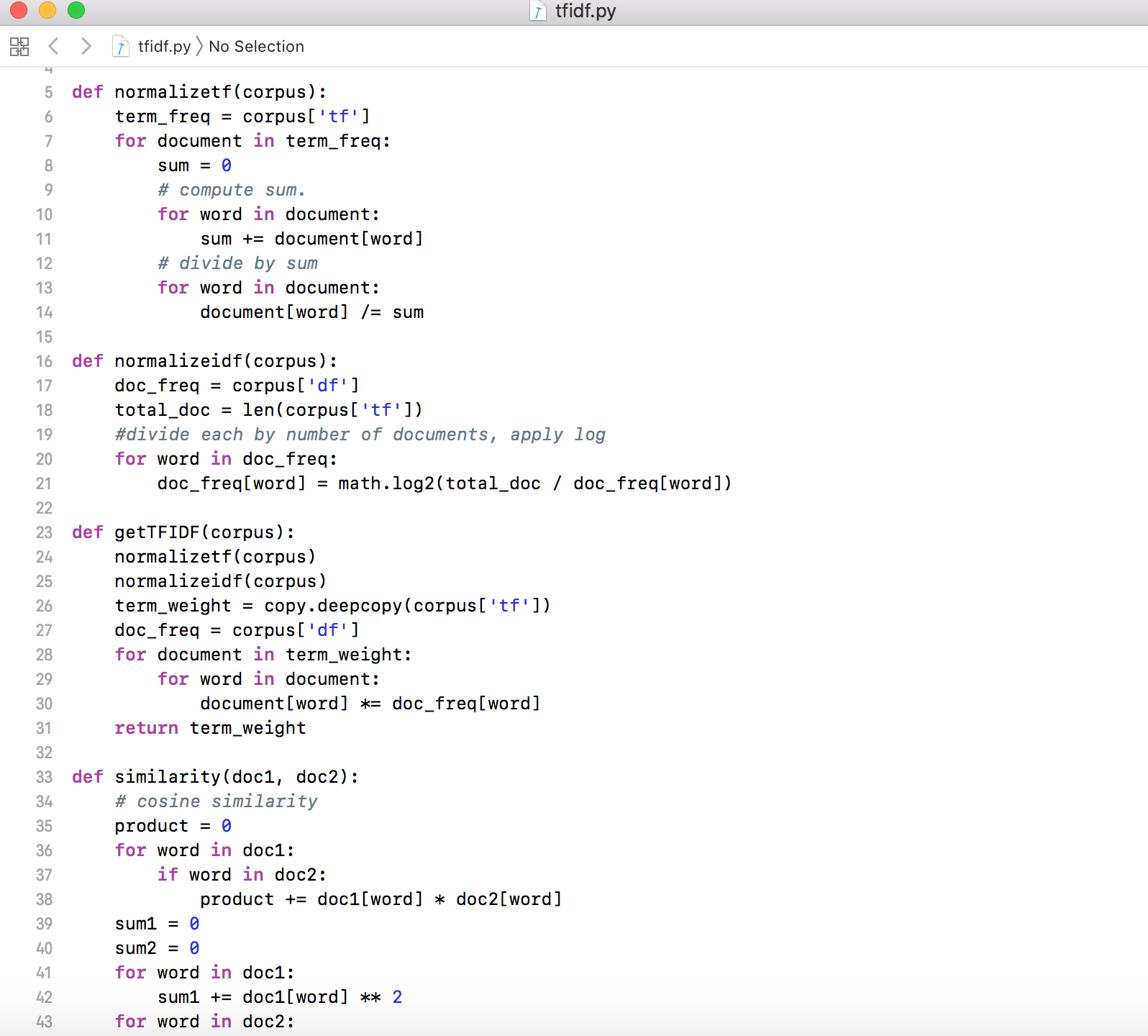
## Tokenization and Stopword Removal

Python NLTK (Natural Language ToolKit) library was used. Default English stopword data was used to remove the stop words. Some of it included: articles (a, an, the) and conjunctions (and, but, if). This step is also combined to remove punctuations, and any other quotation marks.



## TF-IDF weighting

Python dictionary structure is used as term vector representation. Individual news JSON file is aggregated to create corpus with raw term frequency. The term frequency is then normalized based on the total size of the document (proportional normalization). At the same time, document frequency and log2 inverse document frequency is also computed. The whole corpus consists of three parts: term frequency info, document frequency info, and document metadata info (title, url, author, date).



## Query processing

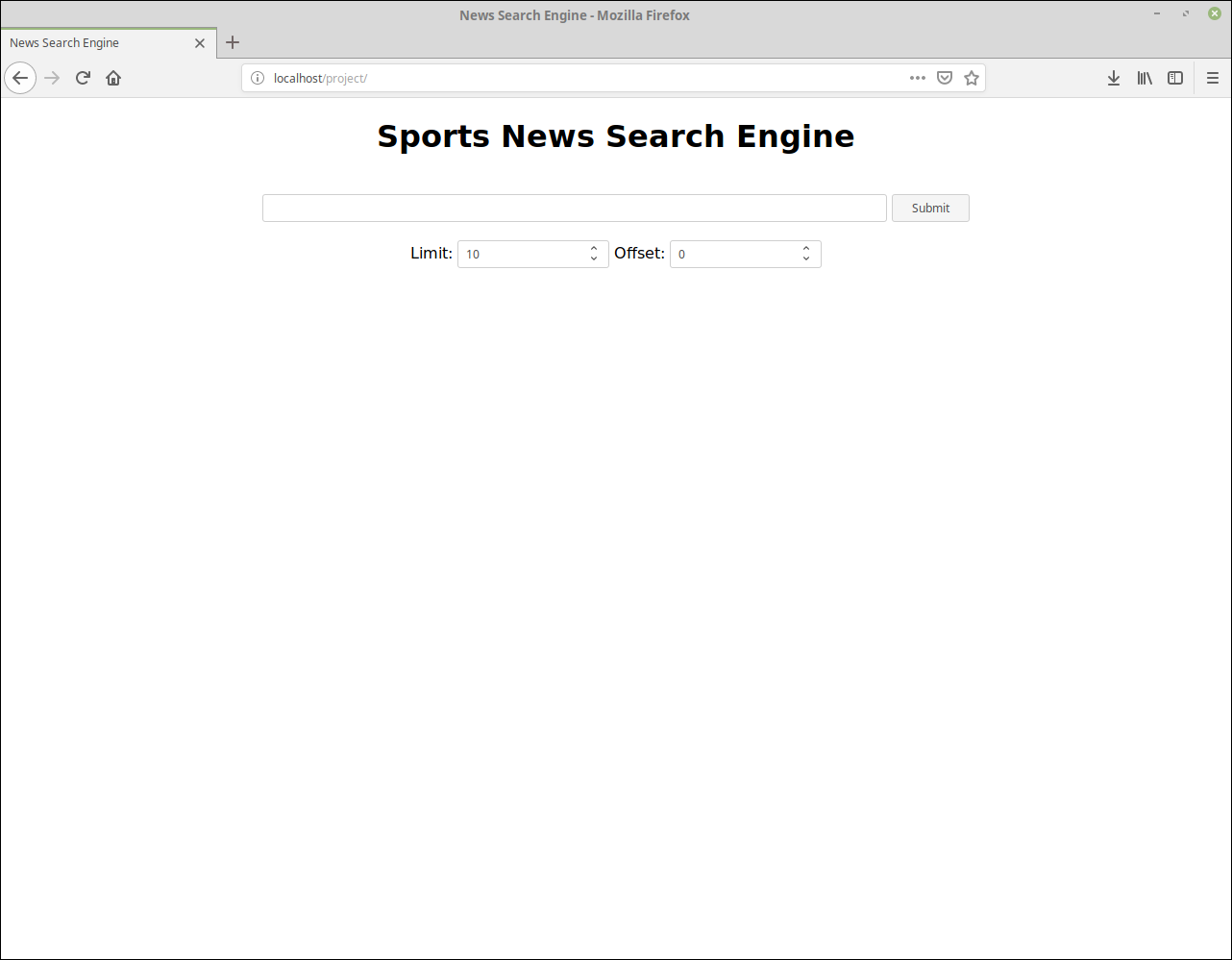
When query is given by user, it goes through exact same process of stopword removal and term weight, inverse document weight processing. The standard cosine similarity measure is used to compute rank of documents. The result is sorted and returned to user.

## Web Interface

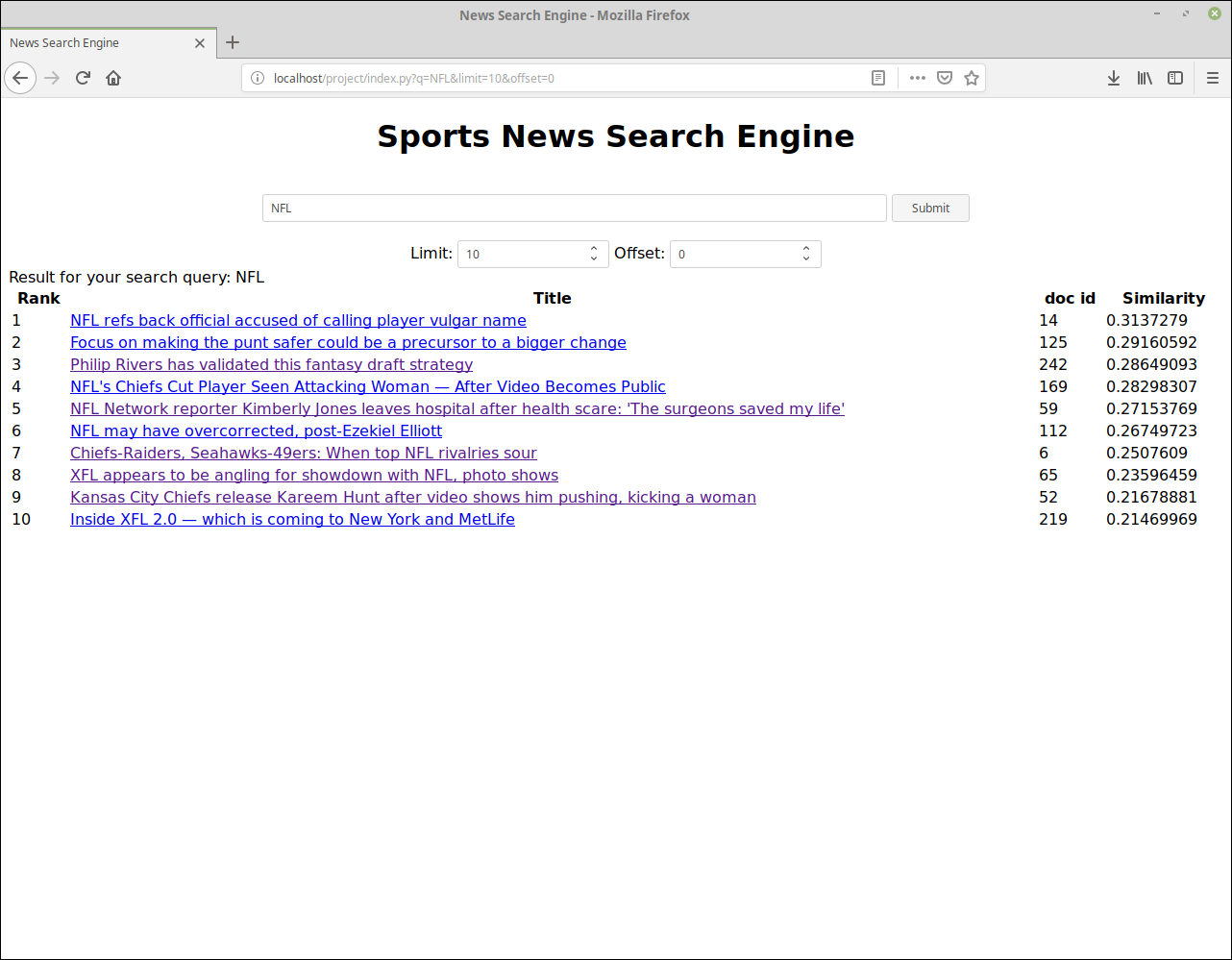
Web interface is created using Common Gateway Interface with Apache HTTP server. Simple text box serves to take user query. There are options to limit the number of document to be displayed, and manual offset to show next set of documents.

# Result

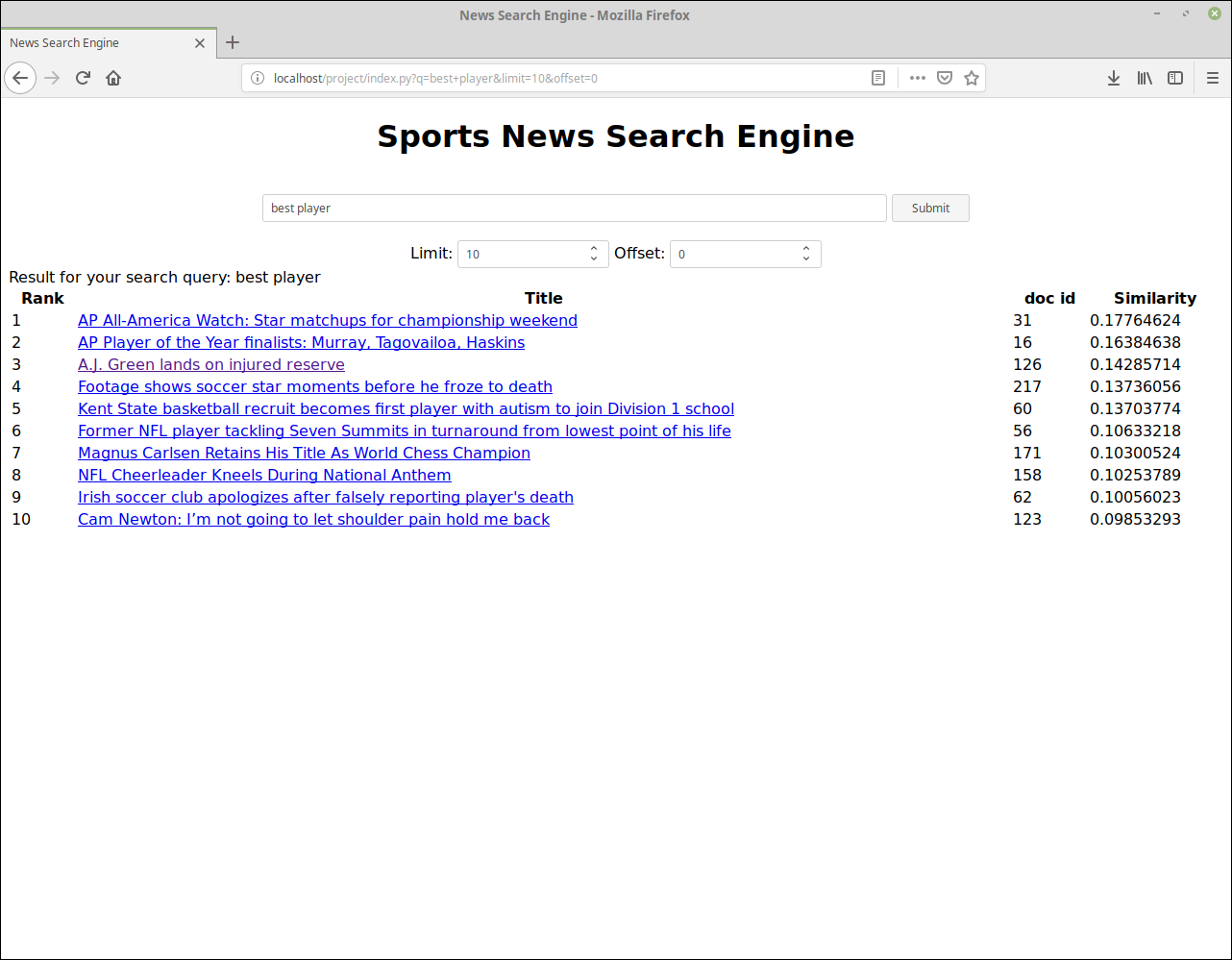
The program will return a list articles with the closest cosine similarity in descending order. The interface will include clickable URL that leads the user to the article.



The webpage is hosted in NYIT’s computer, and can be accessed anywhere inside NYIT campus using following address: (http://192.168.163.215/project/).



Top 10 articles when searched with query “NFL”.



Top 10 articles when search with query “best player”.

# Conclusion

The output of the program has met the expectation of what we proposed. Our goal was to provide relevant articles when the user inputs a query. Overall, the project has completed what the team initially wants but there are room for improvements and more content to be added.

# Contribution

* Hasol: Worked on TF-IDF module, and query module to compute cosine similarity. Created web interface, and configured web server for demo. Created presentation and final report. Responsible for report submission.
* Mansi: Worked with scrapy spiders, collected the data from various websites, HTML Parsing to find the relevant tags. Created presentation and final report.
* Myint: Worked with CSS-based article extraction to retrieve relevant information. Created presentation and final report.
* Qijun: Worked on NLTK library on tokenization and stopwords, reading and writing JSON file. Helped with retrieving data items from websites and query processing. Created presentation and final report.

# Reference

Natural Language Toolkit (https://www.nltk.org/)

Scrapy (https://scrapy.org/)

Apache HTTP server (https://httpd.apache.org/)