System for Recommending News Article to Relevant Patients

Safwan Asghar Abbas, Muddassar Farooq

CureMD, 30 Davis Road, Lahore

Abstract. The concept of web scraping is not new but modern tools have made it easier to crawl through different websites, scrape useful unstructured data and save it for any use. The goal of this project is to build a python based web scraper, design a data pipeline to channel data from different processes and ultimately the data would get displayed to the end user. The web crawlers would collect news articles from reliable online sources and ETL process would be performed on the gathered data. Moreover, this data would get ingested into the raw zone of Data Lake. Then text mining would be performed to extract diseases and prescriptions from the news articles. This data would get stored in MongoDB as it is in an unstructured format. Users would be presented with an interface where they would be required login. After successfully logging in to the website they would enter the name of any disease, ICD10 code for the disease or any NDC. Data such as headings and URLs would be displayed to the user in a fast, efficient, and user-friendly way. This data of news articles would always contain information related to the desired input of user. With the web service built, certain improvements are recommended to make it a finished product ready for live use by all types of users.

Keywords: Web Scraper · Data Lake · Oncology · Hadoop · Apache Kafka · MongoDB · Regex

1 Introduction

1.1 Background and Problem Motivation

With multiple online sources available it is hard to gather all required information from those sources for a single user. And if someone wants to gather information related to their medical condition then multiple issues would occur include is the source providing reliable information and then to shortlist the particular information that the user is interested in. This is an iterative process that consumes too much time and requires a lot of effort which could lead to stress. What if there was a system that would provide this information upon the request of users, this would help them save a lot of time.

A system that has high usability, that is scalable and delivers personalized content to user upon input would help them in finding latest news articles regarding their disease under one platform from multiple resources.

1.2 Overall Aim

The main objective of this project is to build an online web service for users where they would be required to enter the name of disease, diagnosis code or prescription code. Upon their input they would be presented with data scraped from multiple sources. This would contain the headings and URLs of those content which have any information relevant to the input of users. This data would be presented on a website with high usability, users-friendliness, and efficiency.

Python-based web scrapers will be developed to scrape all the available information from multiple reliable sources. This scraped data would get stored into the raw zone of Data Lake. This data will then be extracted, cleansed, and useful data would be extracted which be uploaded to MongoDB. The purpose of using MongoDB is due to unstructured format of data. A dynamic website will be developed where the users can enter disease name and the scraped data will be presented in a user-friendly way.

1.3 Challenges and Limitations

Modern websites are becoming interactive and have dynamic coding. This means information would be displayed upon Ajax calls therefore such websites are not crawler friendly and the concept of static web scraping fails here. Moreover structural changes often occur in websites to improve user experience. Since web crawlers are set up in a way that they scrape data from the elements of web page, so any structural change will bring the web scraper to halt.

Lack of domain knowledge in deep learning resulted in poor understanding of NER models. Due to this reason following with the unavailability of labelled data lead to the usage of pre-trained models for this project (as such models could not be trained). Moreover the ICD10 codes generated from the news articles are not matched to the ICD10 codes of patients that are already present in the database due to other reasons, hence the direction for output has been also changed due to this limitation. Initially it was decided that articles would get displayed to people whose ICD10 code matches with the codes of scraped articles. But now the user will enter the codes or disease name to get desired output.

1.4 Detailed Problem Statement

For this project the following goals have been set:

1. Build Python based web scraper that can scrape information
2. Create a module that ingests the scraped data into the raw zone of Data Lake
3. Apply modified model to extract diseases and prescription
4. Find ICD10 codes and NDC for articles
5. Store data in to a database
6. Build a dynamic website where the data can be displayed for the user from a database

2 Literature Review

In this chapter, all the research that was conducted to complete this project has been presented.

After going through multiple online sources and conducting study it was decided that web scraping would be performed using BeautifulSoup and Selenium. These two libraries will allow to read the content from any web page by sending them a request and fetching the content by loading the pages. This combination not only helps in static scraping but can also be used for dynamic web scraping by using a headless browser [3].

There were many tools available for the streaming of data. Each of them had their own pros and cons, but we had to use an open source tool that was not only free but also scaleable with high fault tolerance. For this purpose after conducting a through study, it was decided to use Apache Kafka as the streaming tool. This is because it can handle large volume of data and is a trusted tool for streaming data [5].

To set up Apache Kafka an online article and online available tutorials were followed [4]. By following those steps a Kafka manager was also setup that would help in visualizing the data or messages streamed, the number of topics and the consumer groups under one platform .

Finally a study was performed to analyze where the final data should be stored. As all of our data is in text form which means it is in unstructured format [6]. Most of the online articles suggest that such data should be stored in NoSQL databases. For this purpose the most suitable database was MongoDB as it was part of the cluster in Data Lake and is easy to use. Therefore MongoDB was selected as the database [2].

3 Methodology

This project can be broken down into different steps with the main goal of building a platform where users can get customized data scraped from different sources. Before displaying the scarped data to the user it must go through a number of stages including cleaning the data, text mining and storing it into a database. After all these steps the data could be presented to the user in a pleasing way. In this chapter, different processes opted for the completion of this project are presented.

3.1 Time Planning

In order to achieve the set goals, the structure and planning of this project are key elements. Three different objects were needed to be built with python based web scrapers, a dynamic website and a data pipeline for ETL processes.

To achieve these objectives a project timeline was created to plan how time was divided between the objectives and report writing.

3.2 Development Process

A development was created to go from the abstract idea of this project to a developed product. The development process for this project can be categorized into the following stages:

Data Collection The initial step was to analyze that what was needed to be done. This means research was required to learn how to build web scrapers. It was also the step where the baseline of the project was outlined. Knowledge regarding static and dynamic web scraping was required.

Data Ingestion Tool Once the data gets scraped from online sources the problem would be to ingest it into the Data Lake. For this purpose a through study was conducted and Apache Kakfa was chosen as the data ingestion tool. This is because it supports real time streaming data. For learning purpose Kafka was set up on Virtual Machine provided by IT department of CureMD and was used throughout the project.

Data Lake As per requirement the data scraped from the internet was supposed to be stored in to the already existing Data Lake of CureMD. As the project was in the initial stages, a single node cluster was setup to store the data. In later stage of the project the address of storage would be shifted to the address of Data Lake.

Text Mining Once the data gets stored into the raw zone, the next task was to extract it from the raw zone and apply some pre-trained model which could possibly extract the terms for diseases and medical prescriptions. For this purpose multiple transformers were used from Hugging Face in an iterative manner [1]. And the ones with better output were selected and ultimately used in the final product.

Labelling Once the text was extracted, certain processes were conducted to ensure that data was clean. Then data related to ICD10 codes and NDC was sourced from the database team to match the diseases and prescription against their codes respectively.

Database All the data scraped, mined and labeled was stored in a form of dictionary. As in there was a single dictionary representing a single article. As all the data was in unstructured format and in the form of dictionaries, this was stored in MongoDB after processing.

User Interface After storing the data into the database, the final step was to present relevant data to the patient. The available frameworks for building the user interface include Django, FastAPI and Flask. For this purpose Flask framework was selected as it is easy to scale and recommended for smaller applications. A searching interface was developed where the user can search any disease, ICD10 code or any NDC. After doing this news articles containing information related to the searched items would be displayed.

3.3 Presentation

During the timeline of this project, an iterative approach was adopted for the development phase to ensure completion of this project in time. After the development process and writing this report, a demo of this project would be presented to the CSO of CureMD.

4 Design and Implementation

This chapter of the report explains how the system was designed, developed and implemented.

4.1 Architecture Diagram

A sketch of how the finished system would work was done early to make sure that everything needed was developed. The architecture diagram for this project is as follows:

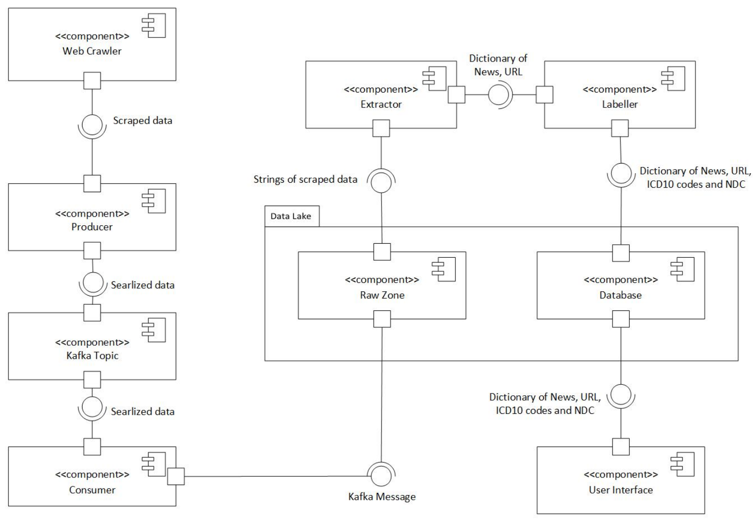


Fig. 1. Architecture diagram showing all the elements that make up the system

1. As shown in the above diagram, a request would be send to the web pages using the web crawler and the text from the page would be extracted
2. A producer created from Apache Kafka would contain the data from python scripts and will push it into Kafka topics
3. Similarly Kafka consumers would be used to extract the messages that were pushed in the topics
4. The consumer would then ingest the data into the raw zone of Data Lake (HDFS) in separate text files for each topic
5. Using Python scripts this data from Data Lake would be extracted and send to the pre-trained models for text mining
6. After extracting the useful text, this text would get labeled against their NDC and ICD10 codes
7. This data would get stored into MongoDB
8. An interface would be used to display the data stored inside the database

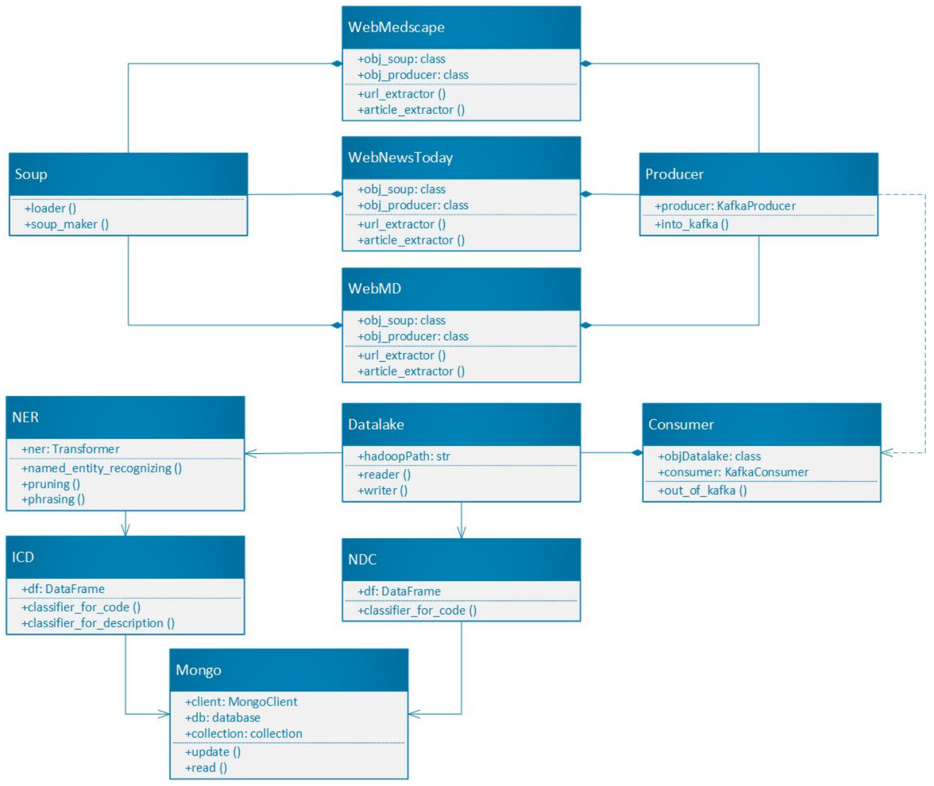
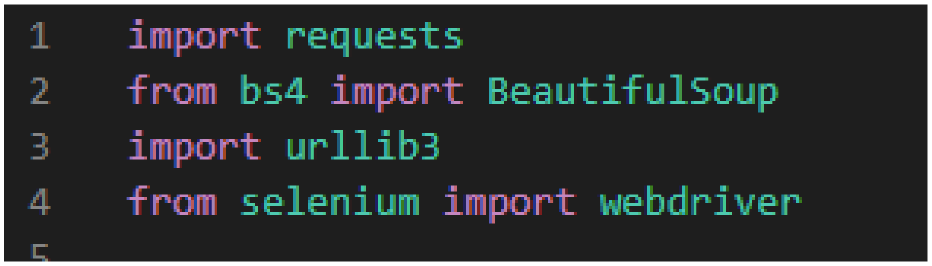


Fig. 2. Class diagram showing all the classes that make up the system

4.2 Building a Python-based Web Scraper

All required libraries were installed on the machine including:

- BeautifulSoup
- Request
- Selenium



```python
import requests
from bs4 import BeautifulSoup
import urllib3
from selenium import webdriver
```

Fig. 3. The libraries used in code

Initially static scraping was performed and the developed web crawlers were able to extract all the desired data from web pages, this included the URLs, headings and the news article itself. But for some web pages it was difficult to apply the same technique.

Therefore dynamic web scraping was required for pages where the request to get the content was denied. This happened because such pages require Ajax calls to load the content hence merely sending request to pages would not work. These web pages were required to get loaded so that the content could be scraped. Therefore the combination of BeautifulSoup and Selenium was applied. Selenium automated the loading of page and the content was accessed using BeautifulSoup.

With the web scraper and using multiprocessing for each web site, it was possible to scrape all the data . And in the end the scraped data was stored in a dictionary format.

4.3 Apache Kafka for Ingestion

The scraped data was planned to be ingested inside the raw zone of Data Lake. For this purpose Apache Kafka was selected as the tool for streaming the data. For learning purpose Kakfa was configured on the Virtual Machine provided by the IT department by the help of an online article.

Following are the components of Apache Kafka:

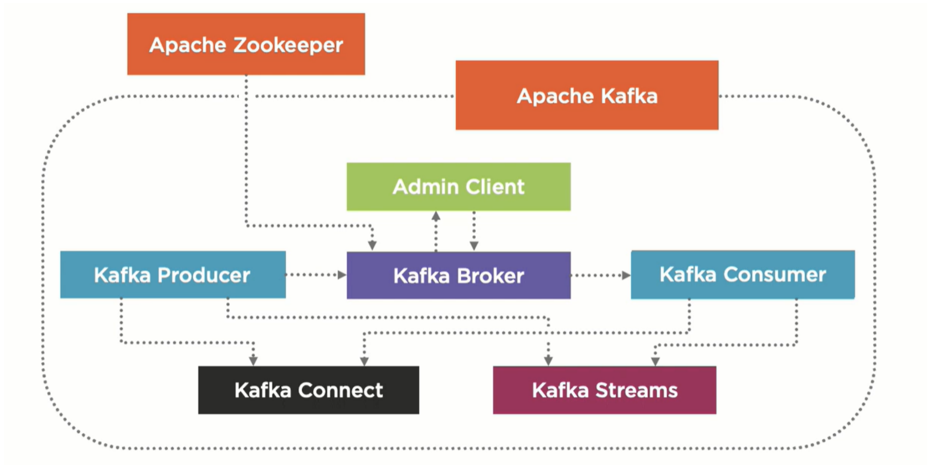


Fig. 4. Kafka Streaming Procedure

Kafka Broker A Kafka broker is a server capable of handling read and write quantities. Each broker has a unique ID and can be responsible for partitions of one or more Kafka topic.

Kafka Topic A Kafka topic is a channel through which data is streamed from one place to another. Topics organize and structure messages, with particular types of messages published inside them. Topics are also identified by unique names and topics can be divided into partitions. Each scraped web site had its own Kafka topic named after it and scraped data for that site was pushed into it.

Kafka Producer A Kafka producer serves as a data source that writes, and publishes messages to Kafka topics. A single producer was used to publish message for scraped data.

Kafka Consumer Kafka consumers belong to group of consumers and are responsible for reading messages from the topics to which they subscribed. Consumers for each website were created to consume message from every topic and store the message in a text file inside the Data Lake.

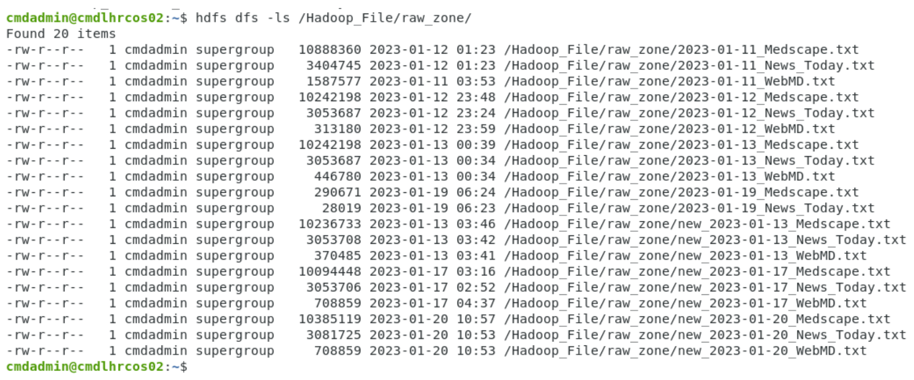
4.4 Raw Zone of Data Lake

A data lake has a flat architecture and may contain many zones depending upon the use cases through which data could generally flow, this including:

– ingestion zone

- landing zone
- raw zone
- processing zone
- consumption zone

In this project there was a constrain to ingest the data into the raw zone. Basically raw zone is the place where data is ingested in its original form without any transformation. Therefore after consuming the data from the Kafka topic, text files were created for current data named after the Kafka topic and the consumed messages were pushed inside the text files in raw zone.



```
cmdadmin@cmdlhrcos02:~$ hdfs dfs -ls /Hadoop_File/raw_zone/
Found 20 items
-rw-r--r--   1 cmdadmin supergroup   10888360 2023-01-12 01:23 /Hadoop_File/raw_zone/2023-01-11_Medscape.txt
-rw-r--r--   1 cmdadmin supergroup    3404745 2023-01-12 01:23 /Hadoop_File/raw_zone/2023-01-11_News_Today.txt
-rw-r--r--   1 cmdadmin supergroup    1587577 2023-01-11 03:53 /Hadoop_File/raw_zone/2023-01-11_WebMD.txt
-rw-r--r--   1 cmdadmin supergroup   10242198 2023-01-12 23:48 /Hadoop_File/raw_zone/2023-01-12_Medscape.txt
-rw-r--r--   1 cmdadmin supergroup    3053687 2023-01-12 23:24 /Hadoop_File/raw_zone/2023-01-12_News_Today.txt
-rw-r--r--   1 cmdadmin supergroup     313180 2023-01-12 23:59 /Hadoop_File/raw_zone/2023-01-12_WebMD.txt
-rw-r--r--   1 cmdadmin supergroup   10242198 2023-01-13 00:39 /Hadoop_File/raw_zone/2023-01-13_Medscape.txt
-rw-r--r--   1 cmdadmin supergroup    3053687 2023-01-13 00:34 /Hadoop_File/raw_zone/2023-01-13_News_Today.txt
-rw-r--r--   1 cmdadmin supergroup     446780 2023-01-13 00:34 /Hadoop_File/raw_zone/2023-01-13_WebMD.txt
-rw-r--r--   1 cmdadmin supergroup     290671 2023-01-19 06:24 /Hadoop_File/raw_zone/2023-01-19_Medscape.txt
-rw-r--r--   1 cmdadmin supergroup      28019 2023-01-19 06:23 /Hadoop_File/raw_zone/2023-01-19_News_Today.txt
-rw-r--r--   1 cmdadmin supergroup   10236733 2023-01-13 03:46 /Hadoop_File/raw_zone/new_2023-01-13_Medscape.txt
-rw-r--r--   1 cmdadmin supergroup    3053708 2023-01-13 03:42 /Hadoop_File/raw_zone/new_2023-01-13_News_Today.txt
-rw-r--r--   1 cmdadmin supergroup     370485 2023-01-13 03:41 /Hadoop_File/raw_zone/new_2023-01-13_WebMD.txt
-rw-r--r--   1 cmdadmin supergroup   10094448 2023-01-17 03:16 /Hadoop_File/raw_zone/new_2023-01-17_Medscape.txt
-rw-r--r--   1 cmdadmin supergroup    3053706 2023-01-17 02:52 /Hadoop_File/raw_zone/new_2023-01-17_News_Today.txt
-rw-r--r--   1 cmdadmin supergroup     708859 2023-01-17 04:37 /Hadoop_File/raw_zone/new_2023-01-17_WebMD.txt
-rw-r--r--   1 cmdadmin supergroup   10385119 2023-01-20 10:57 /Hadoop_File/raw_zone/new_2023-01-20_Medscape.txt
-rw-r--r--   1 cmdadmin supergroup    3081725 2023-01-20 10:53 /Hadoop_File/raw_zone/new_2023-01-20_News_Today.txt
-rw-r--r--   1 cmdadmin supergroup     708859 2023-01-20 10:53 /Hadoop_File/raw_zone/new_2023-01-20_WebMD.txt
cmdadmin@cmdlhrcos02:~$
```

Fig. 5. Files saved after consuming content from Kafka Topics

For testing purpose a single node hadoop file system was set up and the data was pushed inside there.

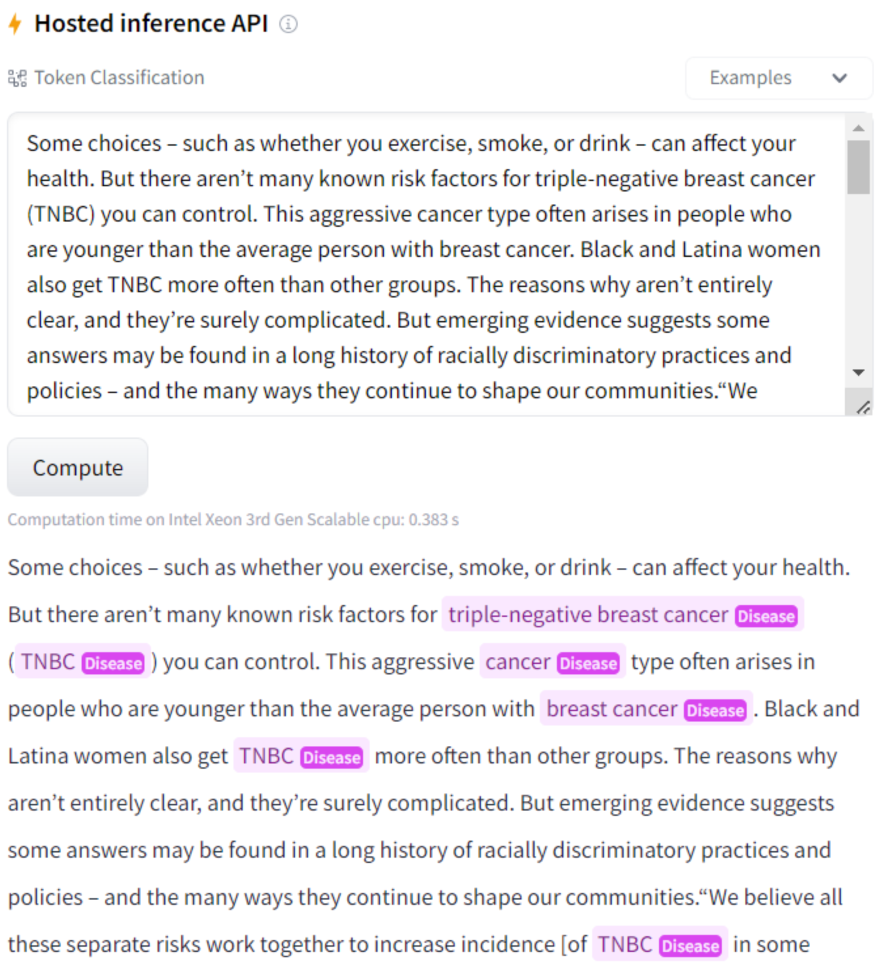
4.5 Extracting Data

Once the data gets completely ingested inside the raw zone, the step is to read the data from Data Lake and apply a pre-trained model to extract useful data from it. Snake-bite library was used to read data from raw zone.

After reading the data, modified bert models were applied on the data. This helped in extracting the name of any disease that could be mentioned in the text. Research was also conducted to find a similar model that could extract names of drugs mentioned in text. But such model was not to be found.

Therefore to extract the name of drugs, regex (a Python library) was used. Although it is a time consuming task but it gets the job done in finding names of medicine. Regex matches the strings for provided scraped text against the NDC library provided by the database team of CureMD.

Similarly the diagnosis codes (ICD10 code) and prescription codes (NDC) were matched for the extracted diseases and medicine names. All of this data was appended to the existing dictionaries of each article.



Some choices – such as whether you exercise, smoke, or drink – can affect your health. But there aren't many known risk factors for triple-negative breast cancer (TNBC) you can control. This aggressive cancer type often arises in people who are younger than the average person with breast cancer. Black and Latina women also get TNBC more often than other groups. The reasons why aren't entirely clear, and they're surely complicated. But emerging evidence suggests some answers may be found in a long history of racially discriminatory practices and policies – and the many ways they continue to shape our communities."We

Computation time on Intel Xeon 3rd Gen Scalable cpu: 0.383 s

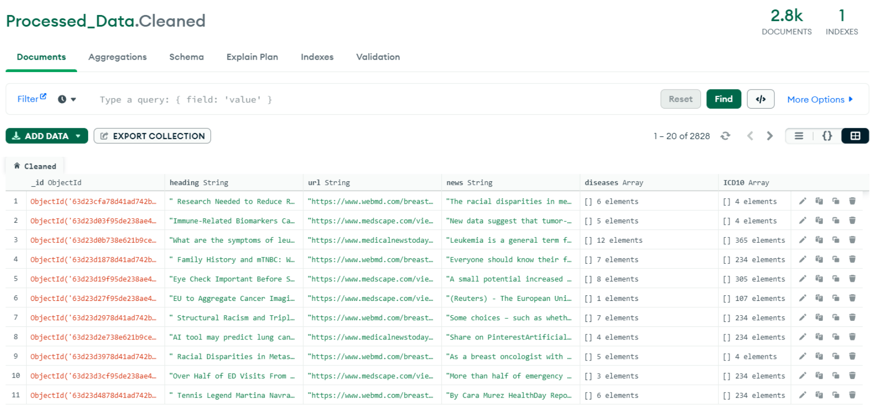
Some choices – such as whether you exercise, smoke, or drink – can affect your health. But there aren't many known risk factors for triple-negative breast cancer Disease (TNBC Disease) you can control. This aggressive cancer Disease type often arises in people who are younger than the average person with breast cancer Disease . Black and Latina women also get TNBC Disease more often than other groups. The reasons why aren't entirely clear, and they're surely complicated. But emerging evidence suggests some answers may be found in a long history of racially discriminatory practices and policies – and the many ways they continue to shape our communities."We believe all these separate risks work together to increase incidence [of TNBC Disease in some

Fig. 6. API from Hugging Face showing results of the used model

4.6 Building a Dynamic Website

Once the scraped data gets collected and required transformations have been performed, this data needs to displayed to end consumer. This calls for the work on building the website where the functions to display the data back to the user exists.

The unstructured data is stored in the MongoDB due to its capability of handling unstructured data in dictionary format. Using flask framework an interface was designed that requires user login and user input to search articles using disease name, ICD10 code or NDC.



	_id ObjectId	heading String	url String	news String	diseases Array	ICD10 Array
1	ObjectId('63d23cfa78d41ad742b…	" Research Needed to Reduce R…	"https://www.webmd.com/breast…	"The racial disparities in me…	[] 6 elements	[] 4 elements
2	ObjectId('63d23d03f95de238ae4…	"Immune-Related Biomarkers Ca…	"https://www.medscape.com/vie…	"New data suggest that tumor-…	[] 5 elements	[] 4 elements
3	ObjectId('63d23d0b738e621b9ce…	"What are the symptoms of leu…	"https://www.medicalnewstoday…	"Leukemia is a general term f…	[] 12 elements	[] 365 elements
4	ObjectId('63d23d1878d41ad742b…	" Family History and mTNBC: W…	"https://www.webmd.com/breast…	"Everyone should know their f…	[] 7 elements	[] 234 elements
5	ObjectId('63d23d19f95de238ae4…	"Eye Check Important Before S…	"https://www.medscape.com/vie…	"A small potential increased …	[] 8 elements	[] 305 elements
6	ObjectId('63d23d27f95de238ae4…	"EU to Aggregate Cancer Imagi…	"https://www.medscape.com/vie…	"(Reuters) - The European Uni…	[] 1 elements	[] 107 elements
7	ObjectId('63d23d2978d41ad742b…	" Structural Racism and Tripl…	"https://www.webmd.com/breast…	"Some choices – such as wheth…	[] 7 elements	[] 234 elements
8	ObjectId('63d23d2e738e621b9ce…	"AI tool may predict lung can…	"https://www.medicalnewstoday…	"Share on PinterestArtificial…	[] 4 elements	[] 234 elements
9	ObjectId('63d23d3978d41ad742b…	" Racial Disparities in Metas…	"https://www.webmd.com/breast…	"As a breast oncologist with …	[] 5 elements	[] 4 elements
10	ObjectId('63d23d3cf95de238ae4…	"Over Half of ED Visits From …	"https://www.medscape.com/vie…	"More than half of emergency …	[] 3 elements	[] 234 elements
11	ObjectId('63d23d4878d41ad742b…	" Tennis Legend Martina Navra…	"https://www.webmd.com/breast…	"By Cara Murez HealthDay Repo…	[] 6 elements	[] 234 elements

Fig. 7. Data stored in MongoDB

Moreover to search through the MongoDB, regex (Regular Expressions) was used as a modifier. This technique can only be used to query strings. For instance, to find the documents where the "diseases" field has a string "can", the regular expression that would be used:

```python
import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")
mydb = myclient["Processed_Data"]
mycol = mydb["Articles"]

#diseases containg can:
myquery = { "diseases": { "$regex": "^can" } }

mydoc = mycol.find(myquery)
```

Fig. 8. Advance querying MongoDB using Regex

Similarly using the above mentioned advanced querying technique the name of diseases, ICD10 codes and NDCs could be filtered from the NoSQL database by taking input from the user at run time. The interface is made using Flask framework due to its simplicity as other frameworks are complex and are recommended for larger applications. Since our interface has a small application therefore Flask is a perfect framework for our use case. The user interface rquires a login from the user to access the portal.

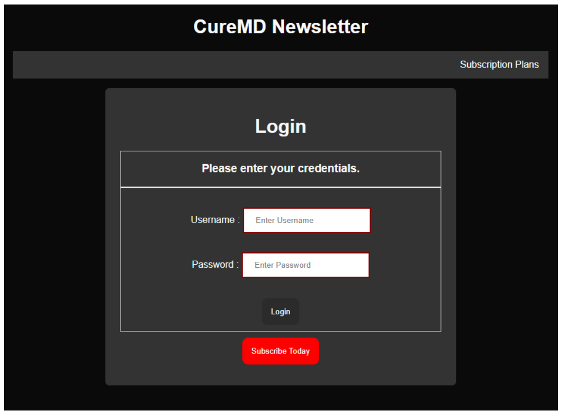
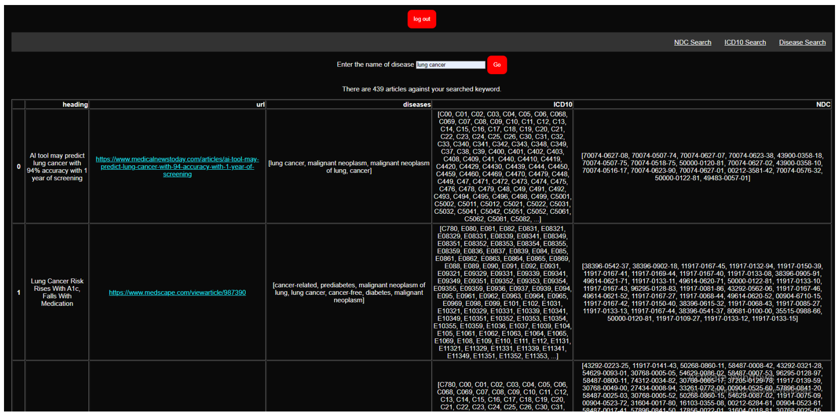


Fig. 9. Login interface of dynamic website

Once the interface was designed and developed, it was connected to the NoSQL database to fetch the data. Once the user enters any thing it gets dynamically creates a query and the query gets searched in the MongoDB. After executing the query results are displayed on the online platform.



There are 439 articles against your searched keyword.

	heading	url	diseases	ICD10	NDC
0	AI tool may predict lung cancer with 94% accuracy with 1 year of screening	https://www.medicalnewstoday.com/articles/ai-tool-may-predict-lung-cancer-with-94-accuracy-with-1-year-of-screening	[lung cancer, malignant neoplasm, malignant neoplasm of lung, cancer]	[C00, C01, C02, C03, C04, C05, C06, C068, C069, C07, C08, C09, C10, C11, C12, C13, C14, C15, C16, C17, C18, C19, C20, C21, C22, C23, C24, C25, C26, C30, C31, C32, C33, C340, C341, C342, C343, C348, C349, C37, C38, C39, C400, C401, C402, C403, C408, C409, C41, C440, C4410, C4419, C4420, C4429, C4430, C4439, C444, C4450, C4459, C4460, C4469, C4470, C4479, C448, C449, C47, C471, C472, C473, C474, C475, C476, C478, C479, C48, C49, C491, C492, C493, C494, C495, C496, C498, C499, C5001, C5002, C5011, C5012, C5021, C5022, C5031, C5032, C5041, C5042, C5051, C5052, C5061, C5062, C5081, C5082, ...]	[70074-0627-08, 70074-0507-74, 70074-0627-07, 70074-0623-38, 43900-0358-18, 70074-0507-75, 70074-0518-75, 50000-0120-81, 70074-0627-02, 43900-0358-10, 70074-0516-17, 70074-0623-90, 70074-0627-01, 00212-3581-42, 70074-0576-32, 50000-0122-81, 49483-0057-01]
1	Lung Cancer Risk Rises With A1c, Falls With Medication	https://www.medscape.com/viewarticle/987390	[cancer-related, prediabetes, malignant neoplasm of lung, lung cancer, cancer-free, diabetes, malignant neoplasm]	[C780, E080, E081, E082, E0831, E08321, E08329, E08331, E08339, E08341, E08349, E08351, E08352, E08353, E08354, E08355, E08359, E0836, E0837, E0839, E084, E085, E0861, E0862, E0863, E0864, E0865, E0869, E088, E089, E090, E091, E092, E0931, E09321, E09329, E09331, E09339, E09341, E09349, E09351, E09352, E09353, E09354, E09355, E09359, E0936, E0937, E0939, E094, E095, E0961, E0962, E0963, E0964, E0965, E0969, E098, E099, E101, E102, E1031, E10321, E10329, E10331, E10339, E10341, E10349, E10351, E10352, E10353, E10354, E10355, E10359, E1036, E1037, E1039, E104, E105, E1061, E1062, E1063, E1064, E1065, E1069, E108, E109, E110, E111, E112, E1131, E11321, E11329, E11331, E11339, E11341, E11349, E11351, E11352, E11353, ...]	[38396-0542-37, 38396-0902-18, 11917-0167-45, 11917-0132-94, 11917-0150-39, 11917-0167-41, 11917-0169-44, 11917-0167-40, 11917-0133-08, 38396-0905-91, 49614-0621-71, 11917-0133-11, 49614-0620-71, 50000-0122-81, 11917-0133-10, 11917-0167-43, 96295-0128-83, 11917-0081-86, 43292-0562-06, 11917-0167-46, 49614-0621-52, 11917-0167-27, 11917-0068-44, 49614-0620-52, 00904-6710-15, 11917-0167-42, 11917-0150-40, 38396-0615-32, 11917-0068-43, 11917-0085-27, 11917-0133-13, 11917-0167-44, 38396-0541-37, 80681-0100-00, 35515-0988-66, 50000-0120-81, 11917-0109-27, 11917-0133-12, 11917-0133-15]
				[C780, C00, C01, C02, C03, C04, C05, C06, C068, C069, C07, C08, C09, C10, C11, C12, C13, C14, C15, C16, C17, C18, C19, C20, C21, C22, C23, C24, C25, C26, C30, C31,	[43292-0223-25, 11917-0141-43, 50268-0860-11, 58487-0008-42, 43292-0321-28, 54629-0093-01, 30768-0005-05, 54629-0086-02, 58487-0007-53, 96295-0128-97, 58487-0800-11, 74312-0034-82, 30768-0005-17, 37205-0129-78, 11917-0139-59, 30768-0049-00, 27434-0008-94, 33261-0772-00, 00904-0525-60, 57896-0841-20, 58487-0025-03, 30768-0005-52, 50268-0860-15, 54629-0087-02, 11917-0075-09, 00904-0523-72, 31604-0017-80, 16103-0355-08, 00212-6284-61, 00904-0523-61, 58487-0017-41, 57896-0841-50, 17856-0022-01, 31604-0018-81, 30768-0025-05

Fig. 10. The final interface of dynamic website

5 Conclusion

In this chapter the results against the developed system have been discussed. The developed system is being evaluated against the goals set out at the start of the project. The developed web scraper has no interface but requires links for web page to be scraped. Once the link is provided the content is scraped and stored. More than 2800 articles have been scraped, ingested into raw zone and after processing they have been stored into the MongoDB. The interface for dynamic website has also been made which has the feature through which users can search through the database to get the desired articles using the Regex advanced querying technique. Recommendations for this project include that customized articles from the database should be shown to the user according to their present medical records in the CureMD database. Such feature will not require the user to enter any input but will display the news after their login.

References

1. https://huggingface.co/models?search=disease
2. Structured vs. unstructured data: What's the difference?, https://www.ibm.com/cloud/blog/structured-vs-unstructured-data#:~:text=Since%20unstructured%20data%20does%20not,%2Drelational%20(NoSQL)%20databases.
3. Web scraping with python using selenium and beautiful soup, https://oak-tree.tech/blog/python-web-scraping-selenium
4. Garg·, G.: How to set up kafka (2019), https://dzone.com/articles/kafka-setup
5. K. Saranya1, S.C., Chelliah, P.R.: Tools and techniques for streaming data: An overview (2020)
6. Raj, S.: Storing of unstructured data into mongodb using consistent hashing algorithm (2015)