



Capstone Project Phase A

News Website Contents Filtering

To browse the news websites in a personalized manner

23-2-D-15



Supervisor : Dr. Natali Levi NatalyL@braude.ac.il

Students: Ronen Zeyan Ronen.zeyan@e.braude.ac.il

Adham Asaad Adham.asaad@e.braude.ac.il

Table of contents

1. INTRODUCTION	3
2. Background and Related Work	5
• web content filtering	5
○ existing filtering methods	5
○ text classification	6
• web crawler	6
• NoSQL	7
3. Excepted Achievements	8
4. Research and engineering process	9
• Process	9
• Product	11
• Diagrams	17
○ Use Case Diagram	17
○ Activity Diagram	20
○ Architecture Diagram	21
○ Class Diagram	22
• GUI	23
5. Evaluation Plan	27
6. References	30

abstract

Our product is focused on the implementation of a filter, provided in a website. The filtering system operates exclusively within the website platform. The filter performs content filtering, from a given news website, based on the user's preferences. The system will enable users to enter a URL address of a website they wish to apply content filtering on, and select undesirable categories or words. For example: filtering all murder-related text. In addition, the system will offer a personalized content filtering and matching. The filtering process involves crawling, extracting, parsing, and classifying web pages. We utilize Python libraries such as BeautifulSoup and Requests for crawling, extracting links, and analyzing HTML pages. In addition, we will employ the Support Vector Machines (SVMs) algorithm for classifying the web page's content of the website into categories. The filtered website, customized to the user's preferences, will be presented on our server using Flask. The main purpose of our product is providing users a tailored browsing experience while maintaining the appearance of the original website.

Keywords

Text classification, news website, WebCrawler, Flask Library, Filtering, customized website.

1. Introduction

In the era of the internet, websites encompass a wide range of content types that are exposed to users. While one part of these contents is interesting and desirable from the user's perspective, the other part is not. Many websites display their content without adapting its content to the desires of a specific user, especially news websites. As a result, the issue we aim to address is how to effectively filter out unwanted content that users do not wish to see, maybe to avoid content that may make them feel uncomfortable, such as content related to terror, riots, sexual content, and more. Therefore, our goal is to implement personalized website filtering that is based on user requests and allowing them comfortably navigate the site. The internet is comprised of two [1] main types of content: textual and multimedia. Textual content includes various forms of text, while multimedia content encompasses images, videos, and more. To regulate access to specific content, different methods of content filtering, commonly referred to as "internet filters," are employed. These filters utilize popular techniques such as browser-based filters, client-side filters, category filters, content analysis, internet service

provider filters, network-based filtering, and search engine filtering [2]. Content filtering systems employ effective approaches such as black/white lists, which maintain records of blocked or allowed URLs, and keyword blocking, which compares keywords to the content in a database and blocks specific content if a match is found. Today, there are diverse solutions for content filtering on the internet, including advanced techniques like web mining and data mining [3].

These techniques facilitate the automatic detection and analysis of information sourced from the web. Furthermore, various tools are available for local content filtering on personal computers or servers. Examples of such tools include K9 [4], Open DNS [5], Dans Guardian and Squid Guard/Squid [6], and Kinder gate [7] parental control. These tools enable page scanning, blocking of unwanted websites, detailed content inspection, and more [8]. In addition to the aforementioned solutions, there are external programs and browser extensions that contribute to content filtering. Programs like Ublock Origin [9] and browser-level filtering in browsers such as Chrome and Firefox allow for URL filtering and blocking. These tools provide an extra layer of control over the content accessible while browsing the internet.

In contrast to the aforementioned solutions, our approach focuses specifically on filtering textual content in news websites. We propose a website that allows users to enter a website URL for filtering. Users will be prompted to specify relevant categories or enter specific words they wish to avoid.

Subsequently, a customized website will be displayed to the user, excluding the unwanted content. Our solution relies on two primary tools: a web crawler [10] and HTML analysis. By Using the web crawler, we automatically scan the requested internet page and extract a list of links to HTML pages from the news website. Additionally, we employ an SVM model to classify each article extracted from the news website links. Articles belonging to undesired categories will be removed from the HTML code, resulting in the display of the news website without the unwanted articles.

Our solution caters to various users, including individuals who seek personalized content filtering on their personal computers, parents who aim to protect their children from inappropriate or sensitive news topics, and even internet service providers who can integrate our filtering system into their offerings. By implementing our news website filter, users can have greater control over the content they consume, ensuring a more tailored and enjoyable news browsing experience. The rest of the paper will be structured as follows: In Chapter 2, we will conduct a literature review and related work analysis, followed by Chapter 3 where we will discuss the expected results before moving on to Chapter 4, where we will describe the engineering process. Finally, we will conclude with Chapter 5, outlining the testing and verification strategy.

2. Background and related work

2.1. web content filtering

Content filtering is the process of screening and restricting or blocking websites based on predefined criteria, such as security concerns or internal content consumption policies. [11]

A web filter, which is a computer software application, plays a crucial role in this process. It analyzes incoming web pages and determines whether to grant or deny permission to view the content. The decision is made by checking the content and its origin against a predefined set of rules.

There are several methods for web content filtering. We will scan the most useful for our work [12] : 1.**whitelists and blacklists** : Blacklists used to block access to specific domains or URLs through a list that the user defined it. while whitelists is used to block all domains or URLs and allow access to specific domains or URLs. The problem of this method is hard to construct and maintain complete and up to date lists. 2.**category filtering** is another method, wherein websites are preassigned to different categories based on their content. Users can then choose which categories to filter and block. 3.**content analysis** used to detect a content or specific keywords and assign a score to each URL. It sets a thresholds and if it exceeded the web page is blocked, the problem of this method is can be easily fail if the specific word changed for example "terror" to te*ror. There are more methods used like DNS based web filtering, proxy filtering, browser based web filtering etc...

By understanding the various methods of web content filtering, we can now shift our focus to the topic of text analysis and classification, which will serve as valuable tools in our project.

2.2. Text classification:

Text classification or categorization is a fundamental process in text mining that involves automatically categorizing documents based on their content. By utilizing various classifiers, such as KNN, NB, SVM, LLSF, Centroid, and Associative classifiers, the aim is to accurately assign documents to their respective classes. This allows for efficient organization, retrieval, and analysis of large volumes of textual data(see figure 1). [13]



In order to filter out unwanted content/articles, we need to be able to analyze and classify the text on each webpage. There are many machine learning algorithms used for this mission such as [14] : (1) **support vector machines (SVM)** represents training documents as vectors and employs hyperplanes to separate different document classes for classification. (2) **K-Nearest Neighbor (KNN)** that uses word vectors to measure similarity and assigns the category of a document based on the categories of its k nearest neighbors in the training set. (3) **Naive Bayes classifiers (NB)** that use word probabilities based on relative word frequencies in documents to assign categories. (4) **Decision Trees** use attribute tests to create a tree structure. Documents are classified by traversing the tree, performing tests at each node, and reaching a final leaf node that assigns the document to a category. There are other algorithms that perform text analysis and classification as well. Each of these algorithms has its own advantages and disadvantages, meaning that some are slower but more accurate while others are faster but less precise and may make mistakes.

2.3. Web crawler

Web crawlers are programs or tools witch called robots, worms or spider that automatically navigate the web, following links between web pages to retrieve and store them. They create a replica of visited pages, Crawlers traverse the internet's graph-like structure, treating web pages as nodes and hyperlinks as edges [10]. Crawlers start with an initial set of URLs, it downloading and extracting links from web pages associated with the URLs. Downloaded pages are stored and indexed, while new URLs are checked for duplicates and assigned for further downloading. This process repeats until all URLs are downloaded. (See Figure 2 for an illustration).

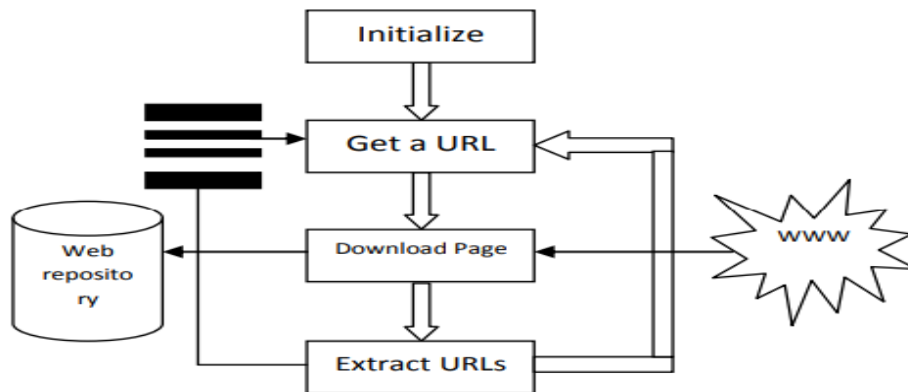


Figure 2: Flow of a crawling process

crawler employs various techniques to optimize its crawling performance. These techniques include three ways: (1) **general purpose crawling** which aims to collect as many pages as possible from a given set of URLs and their links, (2) **focused crawling** that is designed to collect documents only on a specific topic, reducing network traffic and downloads, (3) **distributed crawling** which involves using multiple processes to crawl and download pages from the web.

Now we have discussed web content filtering and web crawling, let's delve into the topic of NoSQL databases.

2.4. NOSQL databases

NoSQL are agile, scalable solutions tailored for handling big data. With flexible data models, they excel in storing and retrieving unstructured or semi-structured data, making them ideal for modern data-driven applications [15]. NoSQL databases offer several advantages over traditional relational databases, including: Fast read and write operations with low latency, they also Scalable for handling large amounts of data and easy to expand and upgrade. Moreover NoSQL offer Lower cost compared to traditional databases. In the other hand NoSQL databases also have some disadvantages, such as not supporting SQL, lacking transactions and reporting features, and being relatively new compared to traditional databases.

While traditional databases use a relational data model, NoSQL databases offer a variety of data models, including : (1) **Key-value databases** store data as pairs of keys and values, allowing for efficient retrieval and high concurrency. Popular examples Redis, Tokyo Cabinet, and Flare. (2) **Column-oriented databases** witch store data by column rather than by row, enabling efficient queries and processing of large datasets.it is ideal for data warehousing and aggregation. Examples include Cassandra and Hypertable. (3) **Document databases** have a similar structure to key-value database but it store data as JSON [16] or XML [17] documents, this providing flexible data

storage and efficient querying. Examples include MongoDB, CouchDB, and Firebase.

2.5. FIREBASE

The Firebase Realtime Database is a cloud-hosted NoSQL database that allows for real-time data syncing across all clients, with offline functionality. Data is stored as JSON [16] format and shared among all connected clients, with automatic updates for the newest data. It requires a fundamentally different data structuring approach than traditional relational databases, utilizing JSON instead of tables and joins. The architectural approach also differs from traditional database. [18]

3. Expected Achievements

Our project aims to achieve specific and measurable outcomes. The main goal is to develop a content Filtering Algorithm that effectively filters out undesired content from news websites based on user-defined preferences. The algorithm will analyze the content of news articles and accurately classify them into desired and undesired categories. In addition, to ensure a seamless user experience, we will implement a User-Friendly Interface. This interface will allow users to easily customize their content preferences and navigate news websites comfortably. Intuitive controls and options will be provided, empowering users to define their own filtering criteria.

Our system will go beyond traditional content filtering by providing a Personalized Browsing Experience. By tailoring the displayed content according to each user's preferences, we will create a more engaging and relevant news consumption experience. Users will have full control over the types of articles they want to see or avoid, enhancing their overall satisfaction.

Alongside these outcomes, our project will address several Unique Features and Challenges. These include incorporating Adaptive Learning techniques into the algorithm to continuously improve and adapt to users' preferences over time. We will also tackle the challenge of handling Multilingual Content, ensuring the system can effectively process and classify articles in different languages. Furthermore, we will address the dynamic nature of news content by developing mechanisms to handle Evolving News Topics and provide up-to-date and relevant filtering results. Finally, we will prioritize Privacy and Data Security, implementing robust measures to protect user preferences and ensure the confidentiality of their personal information.

To evaluate the success of our project, we have established the following Success Criteria:

High efficiency and accuracy in analyzing page content and correct classification: This is the main and central part of achieving a perfectly

working filter. We strive to develop an accurate and efficient filter that can identify and filter out all unwanted content without any errors in selecting the content that needs to be filtered.

Speed of filtering and response: We aim to build a filter that works quickly and efficiently, providing a seamless browsing experience without any noticeable delays caused by the filtering process. We want users to feel like there hasn't been a change in the filtered site's performance.

Ability to update: The algorithm should have the capability to be updated in order to improve the filtering process and adapt to the changing needs of the site and its users. This ensures that the filter remains effective and up-to-date over time.

Positive feedback: Receiving positive feedback from users regarding the correctness and accuracy of the filter will be considered a measure of success. User satisfaction is a key indicator of the filter's effectiveness and impact.

Creating filters that suit the specific user: Developing a filter that can effectively filter content in different languages, designs, and page structures is crucial. The filter should be adaptable and versatile enough to accommodate the diverse needs and preferences of individual users.

By achieving these outcomes, addressing unique features and challenges, and meeting the success criteria, our project aims to deliver a highly effective and personalized content filtering system that enhances the news consumption experience for users.

4. Engineering Process

4.1. Process

Our development process began with manually executing the tasks we wanted to automate. The motivation behind this approach was the need for improved efficiency and scalability. By transitioning to an automated system, we aimed to streamline our operations and achieve our goals more effectively.

To implement our vision, we carefully evaluated different approaches and work methodologies. We selected specific components that were essential to our chosen path.

Throughout the development process, we encountered various theoretical, scientific, and engineering challenges.

These **challenges** were diverse and required thoughtful consideration:

1. User Interface and Data Collection:

Recognizing the importance of a user-friendly interface, we developed a website called "SARR" to facilitate user interaction and gather the necessary information

2. link extraction:

Extracting links from news websites presented a significant challenge. After thorough research, we discovered a viable solution in the form of a "Crawler" tool, which efficiently retrieved the desired links

3.Text classification:

Categorizing articles extracted from links and identifying relevant keywords within the link texts posed its own set of challenges. We explored different models and ultimately chose the SVM model, along with relevant libraries, to train our classification algorithm. Acquiring a suitable dataset for training purposes was a crucial step in this stage.

4. unwanted articles filtering:

To ensure a clean and tailored browsing experience, we incorporated specific libraries capable of removing unwanted articles from the news website.

5. Data Storage and Retrieval:

For storing and retrieving information, we determined that Google's FIREBASE databases, utilizing the JSON format, provided the most suitable solution.

As we move forward to the next stage of our project, we anticipate encountering new challenges, such as website construction and editing. We remain committed to overcoming these obstacles and continuously enhancing our solution to align with our goals and objectives.

The technological constraints that we encountered include:

- 1. Large volume of data:** Dealing with a significant amount of data required efficient methods for data extraction, processing, and classification. Handling large datasets efficiently is a challenge that demands powerful systems and optimized algorithms to ensure quick and accurate processing.
- 2. Runtime and performance considerations:** our process operates on large datasets and at a high pace, thus there will be a need to address performance limitations or runtime constraints. Ensuring efficient

execution and handling scalability become critical factors in delivering timely and reliable results.

3. **provide a suitable dataset for training the model** for each language. This entails challenges such as obtaining sufficient data, ensuring data quality, and aligning with the understanding and usage of the specific language. To overcome this constraint, there may be a need for additional data collection, efficient data processing, and result filtering to deliver a professional and appropriate dataset for training purposes.

By leveraging machine learning algorithms and ready-made tools, we aimed to overcome the technological constraints associated with processing large amounts of data and achieving accurate content classification. These approaches allowed us to efficiently analyze and filter textual content on news websites, providing users a personalized and improved browsing experience.

4.2. PRODUCT

Our product is a specialized website designed to offer personalized content filtering in both Hebrew and English languages. It caters to users who prefer a customized browsing experience on news websites. The main objective is to provide users with the ability to hide articles that are not aligned with their preferences and display only the content that truly interests them. The core functionality of our product is composed of the following components:

First, we receive the website that the user wants to filter.

In the second part, we initiate the filtering process by performing the following operations on the user-provided website:

- a. **"CRAWLING" stage:** We employ a crawler tool to scan the website and extract a collection of outgoing or existing links. To implement this stage, we utilize existing Python libraries such as "BeautifulSoup" and the "Requests" library. Using the Requests library, we send GET requests to retrieve the HTML code of the news website. Then, using the BeautifulSoup library, we scan the HTML code of the site and extract all the hyperlinks while filtering out irrelevant links such as ads. At the end of this stage, we obtain links to all the web pages containing articles found on the requested site. (see figure 3)

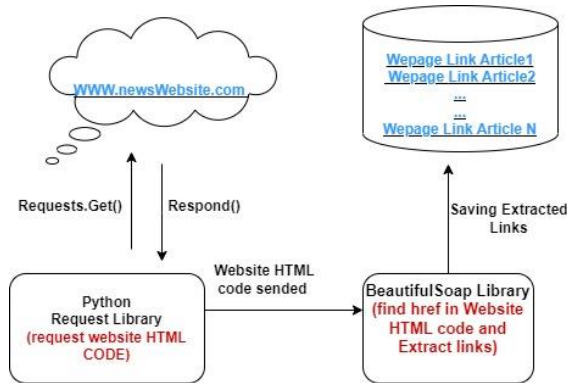


Figure 3: Crawling Process

- b. **"EXTRACTING AND PARSING" stage:** In this stage, we receive the group of links obtained from the crawling process. We use the BeautifulSoup library again to analyze and extract relevant data from the HTML pages. We scan the extracted content, filter out irrelevant elements, and extract clean textual content. This is achieved by utilizing methods in the BeautifulSoup library designed to extract content between specified HTML tags, such as `find()` and `find_all()`. The output of this stage serves as the input for the next stage. (see figure 4)

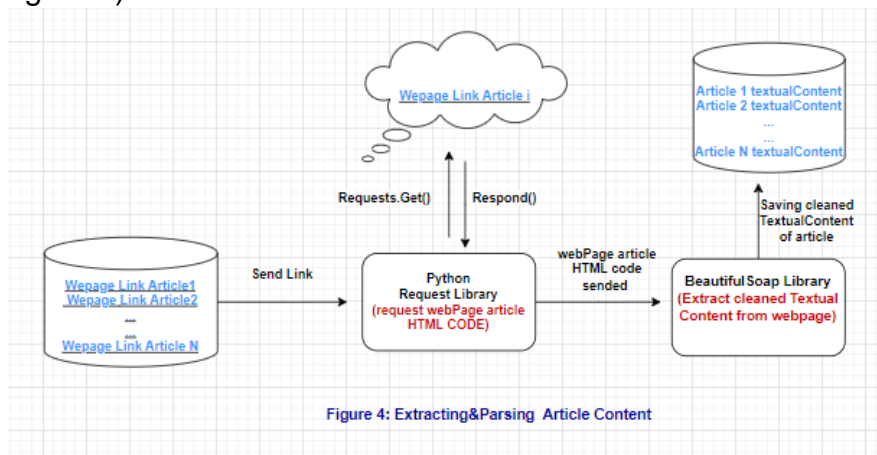


Figure 4: Extracting&Parsing Article Content

c. **"ARTICLES CLASSIFICATION" stage:**

The "CLASSIFICATION" stage is the crucial process in our filtering system. It involves determining the category to which each HTML page belongs. The data obtained from the previous stage represents the textual content of each HTML page from the desired site to be filtered. In this stage, we input the data (articles content) into a classification algorithm that analyzes the content of each page(cleaned textual content) and assigns it a category. We also store the links of the pages classified into unwanted categories for later use in presenting the filtered news website.

In our project, users have two options for filtering. They can choose categories they are not interested in seeing content from, or they can enter specific words to filter content containing those words. For example, if a user selects the "Politics" category, all the articles in the website related to politics will be filtered out. Similarly, if a user enters the word "murder," all articles discussing murder will be filtered.

After conducting research and analyzing various classification algorithms, we decided to utilize two algorithms, one for categories and another for words. For category-based filtering, we sought an algorithm that is efficient and fast in classifying web pages into their appropriate categories. Ultimately, we chose to use the Support Vector Machine (SVM) algorithm. The decision to select SVM was based on its success in achieving good and consistent performance in text classification tasks compared to other algorithms. Additionally, SVM offers features such as speed and accuracy, making it an efficient and straightforward model for the classification task. [19]

SVM is a supervised machine learning algorithm used for both classification and regression problems. It aims to find an optimal hyperplane that maximally separates different classes in the dataset. By default, SVM is a binary classification algorithm, but it can be extended for multiclass classification using techniques such as One vs One (OVO), One vs All (OVA), and Directed Acyclic Graph (DAG). In OVO, the problem is divided into pairwise binary classification tasks, and the final prediction is determined through majority voting. OVA trains a binary classifier for each class against the rest, and the class with the highest confidence is predicted. DAG constructs a graph of binary classifiers, and predictions are made by traversing the graph. These approaches enable SVM to efficiently handle multiclass classification problems. [20]

We will employ the OVA technique, which means there will be N models of SVM, one for each category. Each model will determine whether an article belongs to its corresponding category or not. The model with the highest confidence score will be selected.

During the training process, we will provide the SVM models a set of different texts (documents) along with their corresponding categories (labels). Since a single SVM can only work on one language, we will have multiple pre-trained SVMs, one for each language we support. By using this model and the training dataset, we will classify the articles into the appropriate categories as defined. (see figure 5)

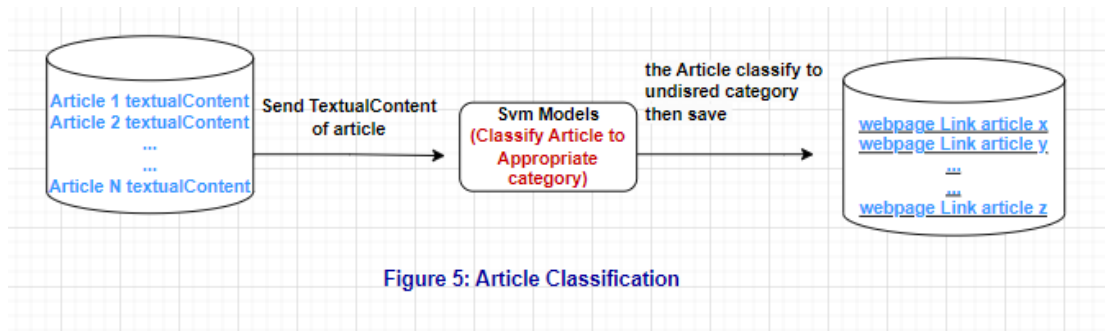


Figure 5: Article Classification

For word-based filtering, we will convert the received words to their stems. For example, if a user enters the word "murdering," the stem of the word will be "murder." We will also convert the article's content to the stems of its main words. Finally, we will search for the appearance of the received words in the converted article. If the words appear in an article, the webpage of that article will be filtered. To accomplish this, we will utilize the NLTK (Natural Language Toolkit) Stem library.

By implementing these classification techniques, our filtering system can accurately categorize website content and filter articles based on user-defined categories or specific keywords.

In the final part, we present the filtered and customized website to the user. This part consists of two stages:

a. Editing the website's HTML code: During this stage, we manipulate the HTML code of the website. Using information obtained from the classification stage, we identify and delete the parts of the HTML code that contain links to unwanted web pages. The BeautifulSoup library facilitates this process by parsing the HTML code, locating the targeted elements, and eliminating them. The result is a modified version of the news website HTML code without the unwanted articles.

b. Displaying the evaluated HTML code on our server: In this stage, we build a website that is a copy of the original website the user wants to filter. We employ the Flask library [21] to create a server that displays the evaluated website's code. The user can access and view the filtered website on our server. The modified HTML code retains the original appearance of the site, and the filtering process is transparent to the user.(see figure 6)

By implementing these stages, our product enables users to browse news websites in a personalized manner, tailored to their preferences, and without the presence of unwanted content. (see figure 7)

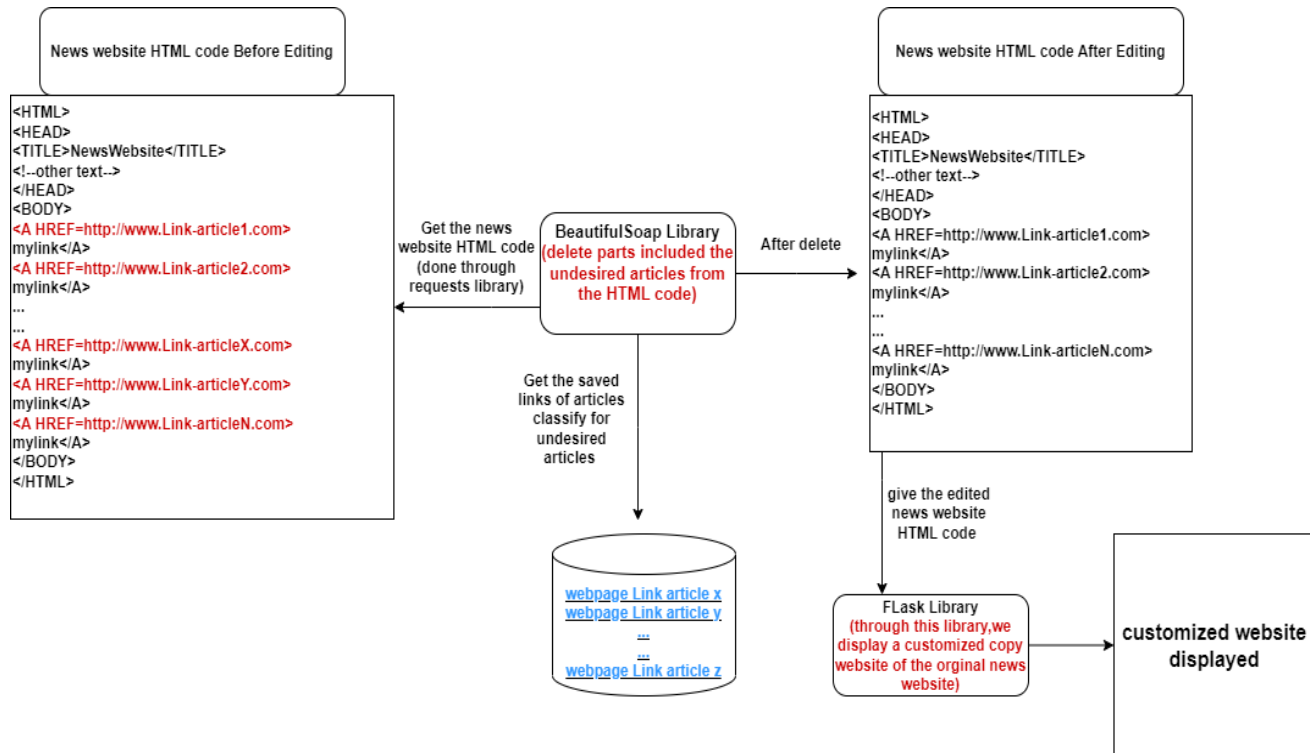


Figure 6: Editing and display the news website

This diagram summarizes all the processes we described

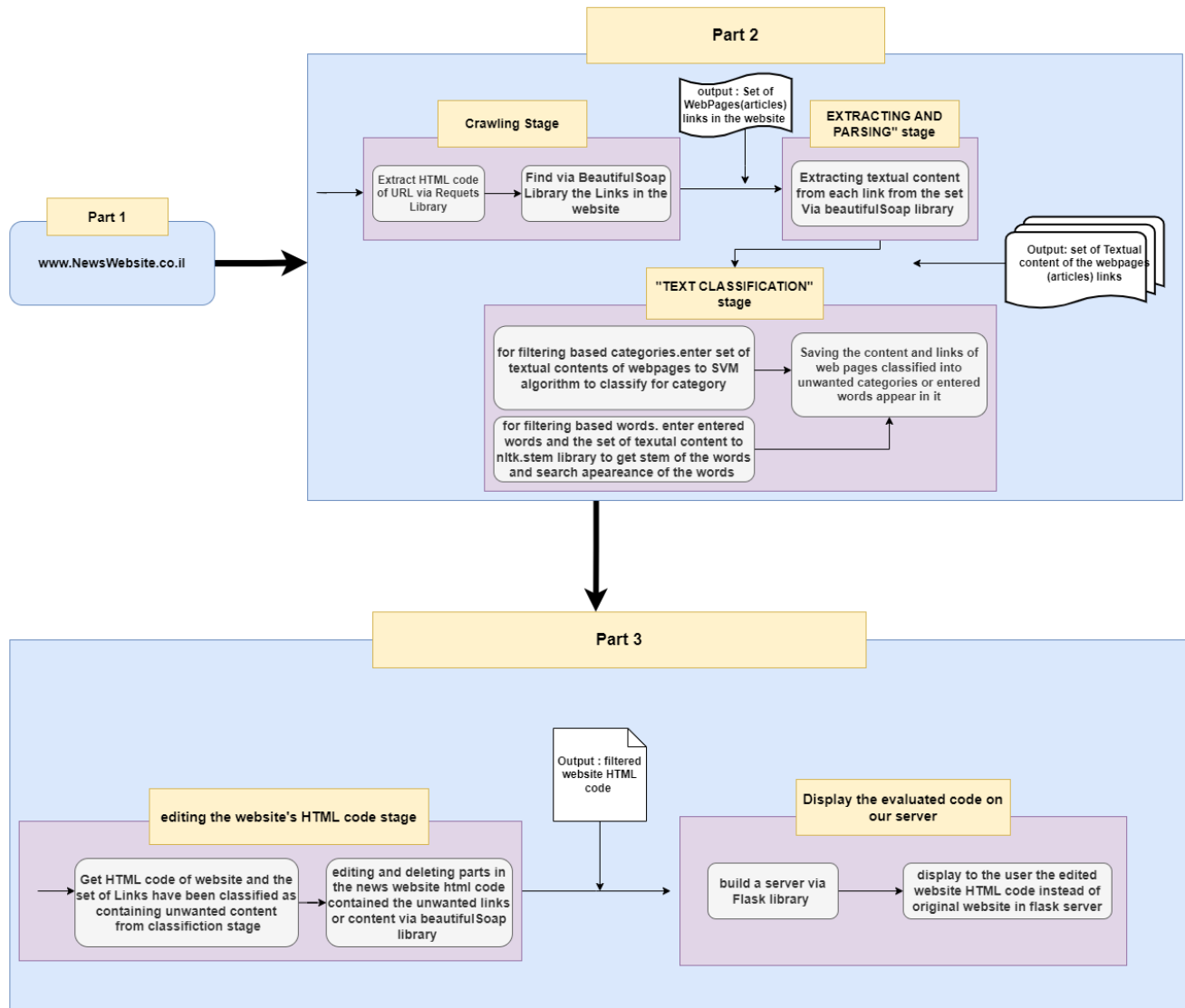
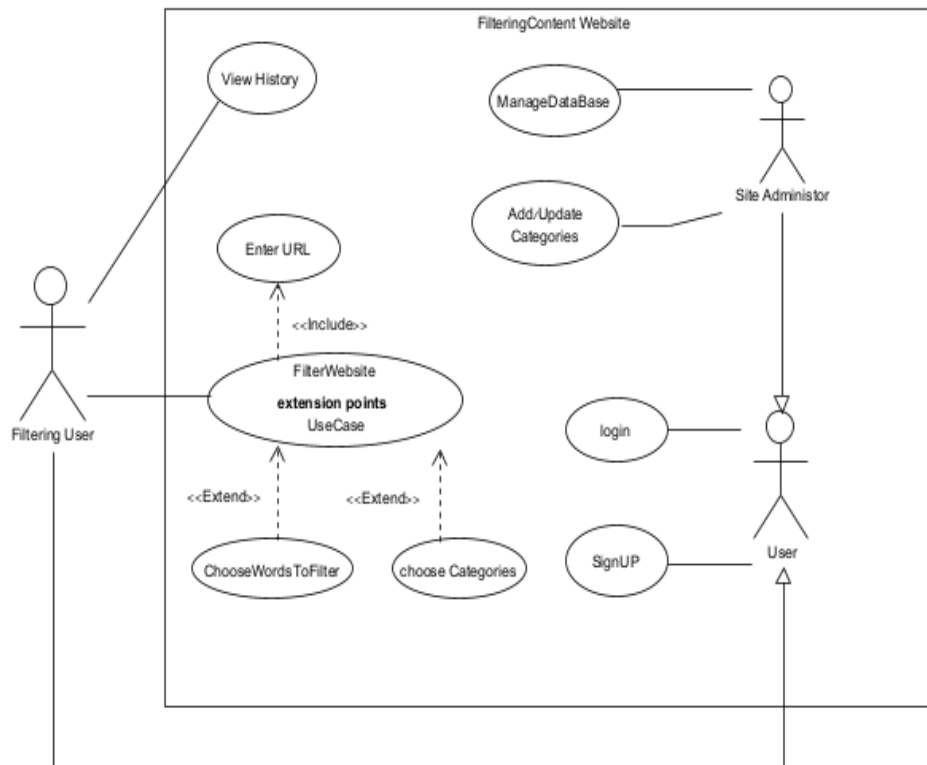


Figure 7 : summarize of Our algorithm

Use Case:



Use case	Filter Website
Actors	Filtering User
Trigger	User press "Generate Filtered website"
Pre-condition	User Sign-up and login
Post-condition	Filtering website displayed
Successful scenario	<ol style="list-style-type: none"> 1. The user enters a URL to be filtered. 2. The user chooses categories that he do not want to see. 3. The system checks the validity of the website and displays a message indicating the filtering process. 4. The system crawls the web and extracts links to all HTML pages on the website. 5. The system scans all the pages and extracts relevant content from each page and prepares it. 6. The system analyzes the prepared content using the selected algorithm. 7. The system displays to the user the filtered and prepared website ready for browsing.

	<p>8. The user clicks to view the filtered website.</p> <p>9. The system displays a copy of the filtered website to the user.</p>
Alternative scenario	<p>1. The user enters an invalid URL.</p> <p>2. The system displays an error message and prompts the user to verify the URL's validity.</p> <p>3. The system redirects the user back to the homepage to enter a valid URL.</p>

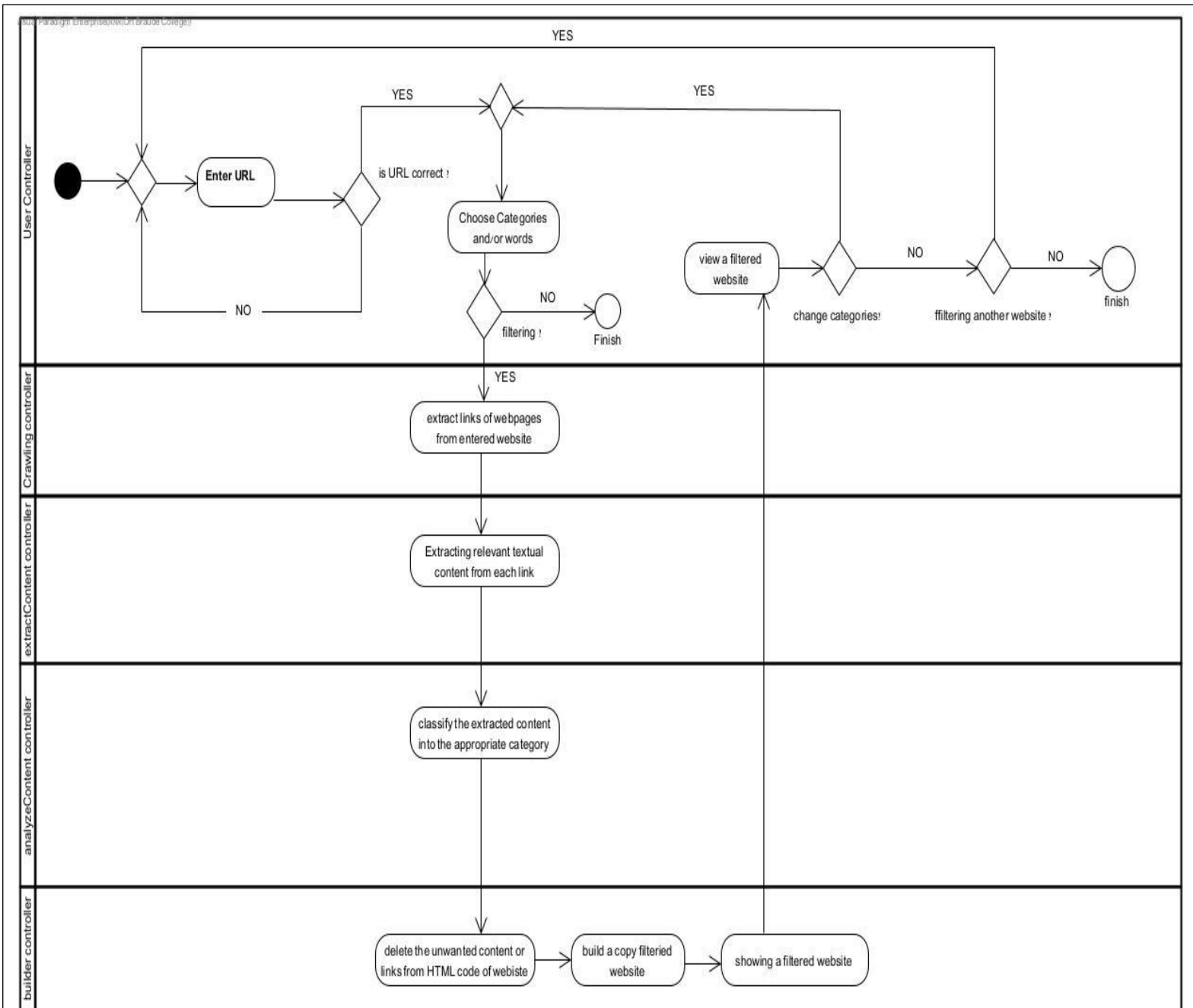
Use case	View History
Actors	Filtering User
Trigger	User press "View History"
Pre-condition	User Sign-up and login
Post-condition	System Displays a History of filtered sites and selected categories selected by the user
Successful scenario	<p>1.The system displays a list of websites that were previously entered and the selected categories or related words.</p> <p>2.The user selects a website from the list.</p> <p>3.The system requests confirmation if there are any changes in the selected categories.</p> <p>4.The system performs filtering.</p> <p>5.The system displays the filtered website to the user.</p>
alternative	1. The system displays an empty list of websites.

Use case	Manage Database
Actors	site administrator
Trigger	User press " Manage Database"
Pre-condition	User Sign-up and login as site administrator
Post-condition	The database changed at the administrator request
Successful scenario	<p>1.system display "import" & "export" buttons.</p> <p>2. user presses "import" button.</p> <p>3. The system displays a screen with attach file selection rectangle.</p> <p>4. The user choose file wanted to upload such as new documents for training SVM model for new categories etc...</p> <p>5. The User presses "upload and save" button.</p> <p>6. The System added the uploaded file to the database.</p>

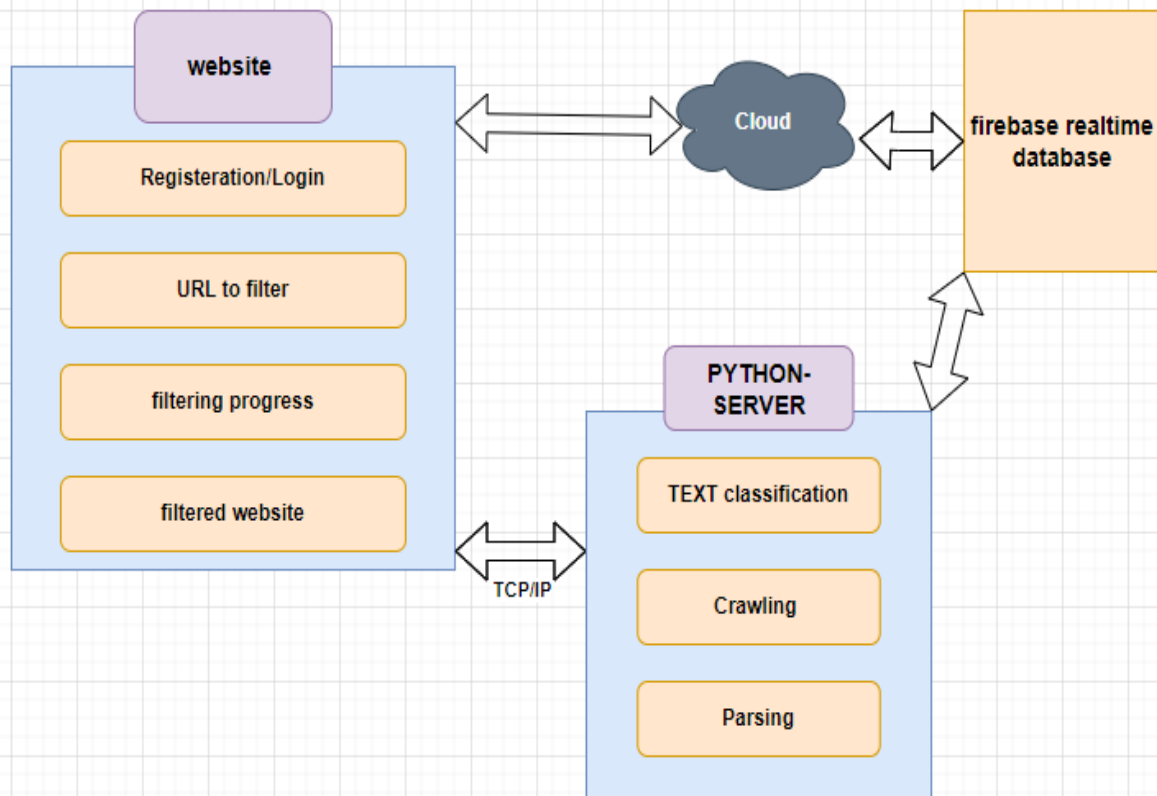
	7. The System shows a "file saved successfully" message.
alternative	6.1 The System found an error while saving the file. 6.2 The System display an error message "error saving file".

Use case	Add/update Categories
Actors	site administrator
Trigger	User press "add/delete/update categories"
Pre-condition	User Sign-up and login as site administrator
Post-condition	new categories have been added
Successful scenario	1. The system presents a list of categories and displays buttons for adding/deleting/editing. 2. The user clicks on "Add". 3. The system displays input field for entering the category name 4. The user enters a new category name that does not exist and press Update&save. 5.system saved the new category 6. system display message "saved successfully" and finish button
alternative	4.The system displays an error indicating that the "category already exists".

Activity:



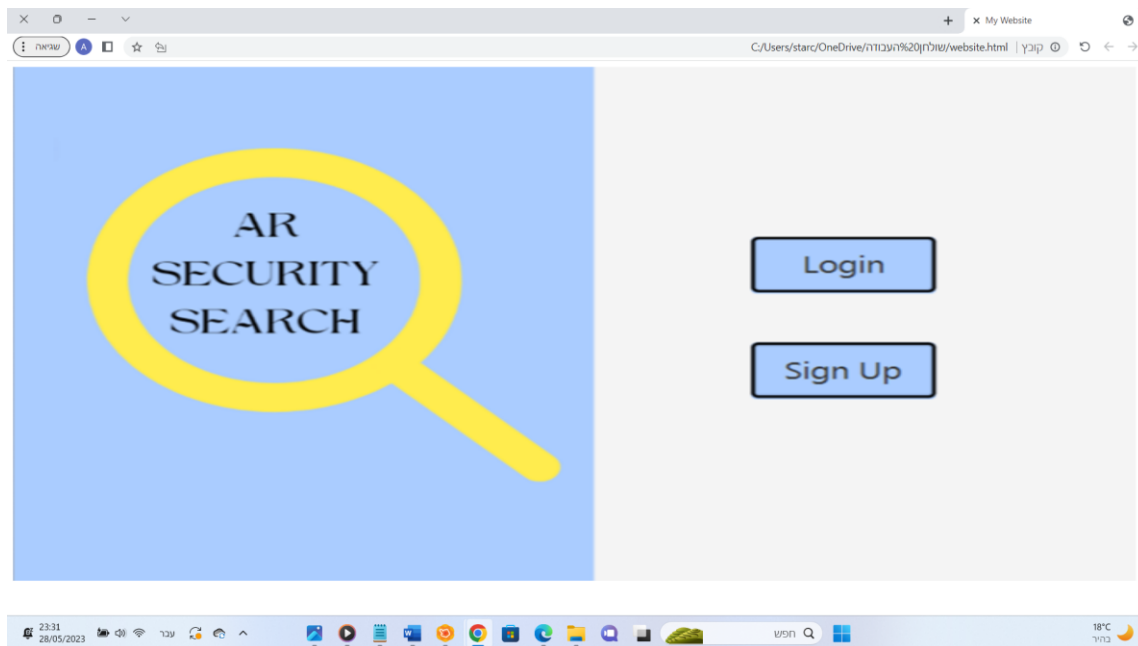
Architecture diagram:



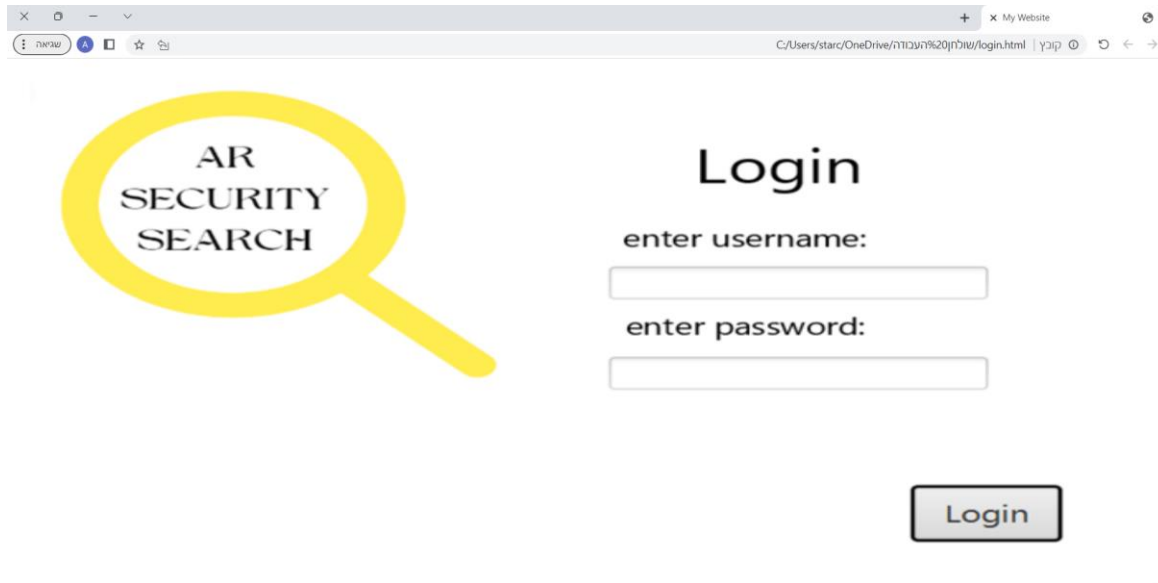
GUI

When opening the website will display this page with two option

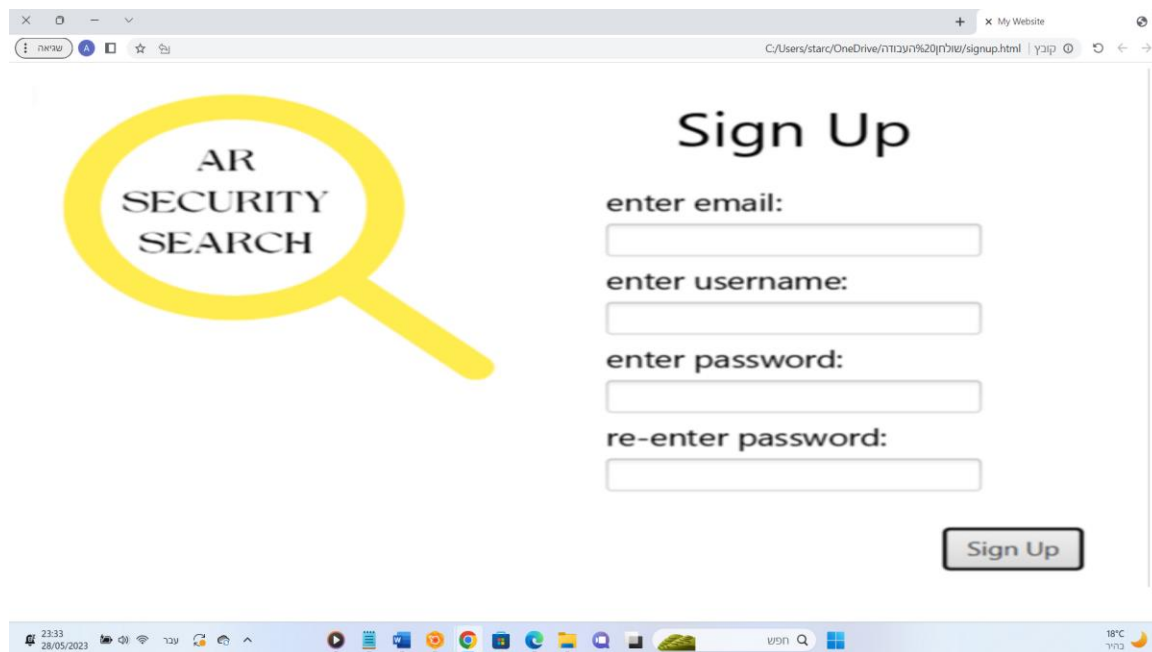
- 1). **Login:** To connect using a username and password
- 2). **Sign Up:** To register on our website and create an account.



If you click on "**Login**" in the main window, you will receive the window in front of you. In this window, you will connect using the username and password.



If you click on **"Sign Up"** in the main window, you will receive the window in front of you. In this window, you will register for the website by entering your email, username, and password twice to ensure that there are no errors in the password entered and didn't noticed.

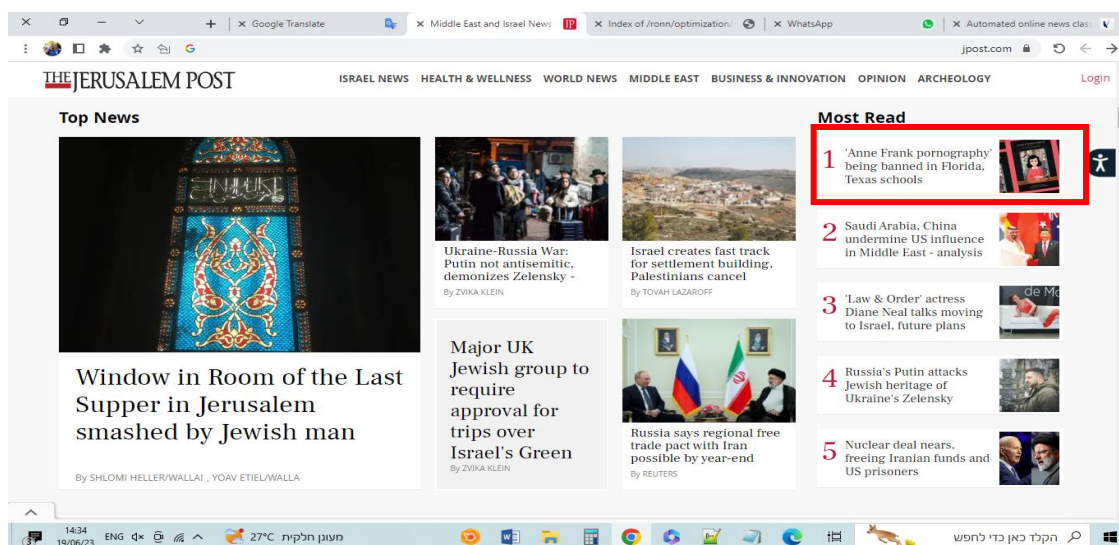


After completing the registration process or when logging in directly, will display this window, which is the most important window on our website as it allows you to filter the content. Through this window, you can select categories to filter the websites based on them, or you can enter specific

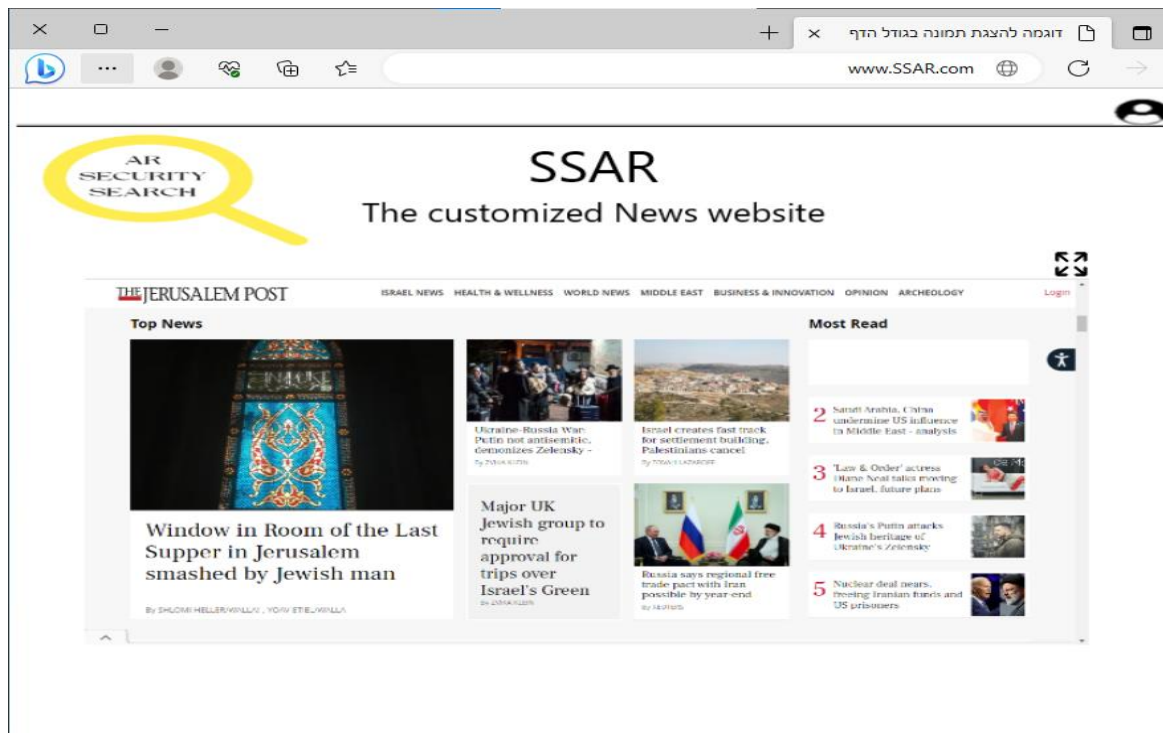
keywords to filter articles that contain them or words with the same root. After clicking the **"Filter"** button, you will receive the filtered website. Additionally, there is an option to view the history of the last 10 searches through the "View History" button.

In this window, suppose the user wanted to filter the news website, so he enters the site in the "Enter website:" box <https://www.ipost.com/> And choose the category of SEXUAL. then every article related to the SEXUAL category is filtered.

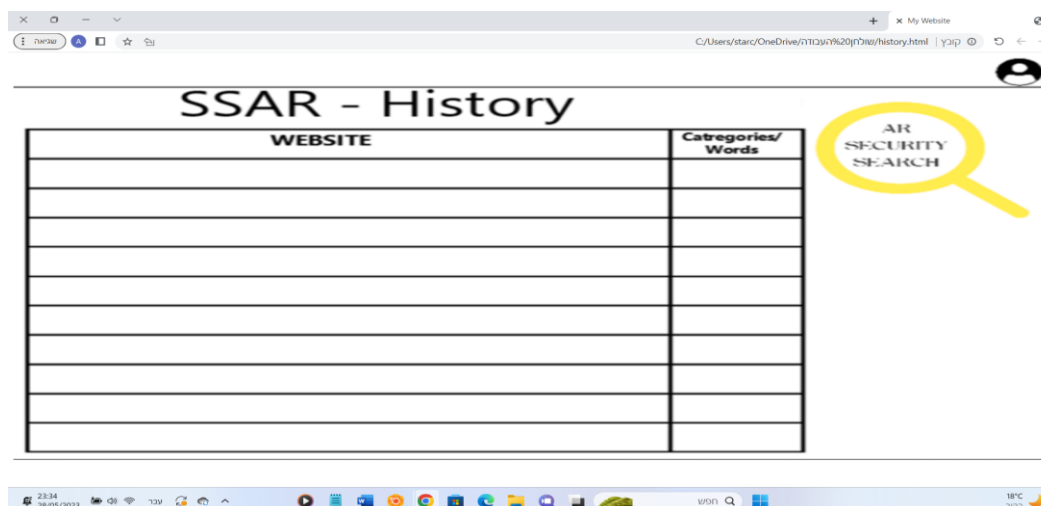
This is the website before filtering (the article in red box related to sexual category)



this is how the news website displays in our website after filtering.



In this window, the history of the last 10 searches is displayed. The information will display to you is the website and the words or categories that were used to filter the website. (When going back, you will receive the filtering window again.)



5. Testing Plan

Our product offers content filtering capabilities for news websites in two languages: Hebrew and English.

To ensure the proper functioning and alignment with our expectations, we have designed a testing approach incorporating the following methods:

5.1. Evaluation by Users:

We will engage eight volunteer participants to assess our website and provide valuable feedback. Four users will evaluate the filtering system using predefined categories, while the other four users will test the filtering system using free keywords. Each volunteer will be assigned a different structured news website in a distinct language.

The objective of this evaluation is to gauge the effectiveness of our website in filtering the requested content and to evaluate the speed at which tasks are executed.

Upon completion of the testing phase, we will meticulously analyze the results obtained from both user groups. If any issues surface, we will conduct a thorough examination to identify areas requiring improvement, with the aim of enhancing the functionality and speed of our website.

5.2. Unit Testing:

In order to ensure that our website processes are functioning smoothly and to verify the reliability of our final product, we have implemented a thorough testing plan. We have divided the tests into specific cases and for each case, we are conducting unit tests and outlining the test's purpose. For every test, we have written a detailed description of the expected results. These descriptions and the test results can be found in the table below:

Test Number	Case	Purpose	Headline	Expected Results
1	Sign Up and Login	To guarantee that the inputted information is accurately	Sign Up.	Go to the login screen after creating new user in the database.

2		processed and securely stored.	Login.	Go to the home screen after connecting to the user in the database.
3			Failed sign up.	Highlight the error.
4			Failed login.	Highlight the error.
5	Crawling	To make sure that we extract all the existing links of the provided website.	Crawling successfully.	It will send the existing links to the next part (parsing).
6	Parsing	We will scan the extracted content and extract relevant and clean textual content from the links we received.	Parsing successfully.	Send the texts we received in each link to the next stage (Classification).
7	Classification	we essentially determine the category to which each HTML page belongs.	Classification successfully.	Send the categories to our website to compare with the unwanted categories that the user choose.
8	Algorithm	Ensure proper functioning of SVM.	Process text using SVM-text related to undesired category.	Classify to the undesired category .

9			Process text using SVM-text related to desired category.	Classify to the desired category .
10	Free words filtering	Ensure proper functioning of filtering based free words.	User enter free words to filtering all articles related to this words	Filtering successfully all the articles related to this words
11	New URL	Receive the URL of the website after filtering.	User receive new URL to the filtering website.	It will displayed on the screen of the filtering.
12	History	Make sure that the history of the last searches was saved.	The searches existed.	The user see information about the last ten searches he does.
13			The searches not existed.	The user see empty list of searches.

6. References

- [1] "webcontentType," [Online]. Available: <https://www.techopedia.com/definition/23885/web-content>.
- [2] V.K.T.Karthikeyan, "Web Content Filtering Techniques," vol. 5, pp. 203-208.
- [3] D. J. Hand, "Principles of Data Mining," vol. 30, no. 7, pp. 621-622, 2007.
- [4] "k9 web protection," [Online]. Available: https://en.wikipedia.org/wiki/K9_Web_Protection.
- [5] "OpenDNS Parental Controls," [Online]. Available: <https://www.cd-csd.org/wp-content/uploads/2017/07/OpenDns-Parental-Controls.pdf>.
- [6] S. Saxena, "Content Filtering using Internet Proxy Servers," vol. 3, no. 2, pp. 10-15, 2013.
- [7] "What is KinderGate Parental Control?," [Online]. Available: <https://www.kindergate-parental-control.com/product/overview>.
- [8] D. S. K. Ankur Baishya, "A Review on Web Content Filtering, Its Technique and Prospects," vol. 7, no. 3, pp. 37-40, 2019.
- [9] [Online]. Available: <https://ublockorigin.com/>.
- [10] V. S. D. K. S. Md. Abu Kausar, "Web Crawler: A Review," vol. 63, no. 2, pp. 31-36, 2013.
- [11] "webcontentfiltering," [Online]. Available: <https://nordlayer.com/blog/what-is-content-filtering/>.
- [12] "methodsforFiltering," [Online]. Available: https://perception-point.io/guides/browser-security/web-filtering-an-in-depth-look/#Content_Filtering_Methods.
- [13] V. Korde, "TEXT CLASSIFICATION AND CLASSIFIERS:A SURVEY," vol. 3, no. 2, pp. 85-99, 2012.
- [14] R. S.-N. S. R. Du, "Web filtering using text classification," Vols. 325-330, 2003.
- [15] H. E. G. L. D. Jing Han, "Survey on NoSQL Database," pp. 363-366.
- [16] P. Y. H. Teng Lv, "Survey on JSON Data Modelling," pp. 1-5, 2018.
- [17] X. W. . D. Theodoratos, "A survey on XML streaming evaluation techniques," pp. 178-202, 2012.
- [18] "Firebase Realtime Database," [Online]. Available: <https://firebase.google.com/docs/database>.
- [19] T. Joachims, "Text Categorization with Support Vector Machines: Learning with Many Relevant".

- [20] "svm multiclass classification," [Online]. Available:
<https://www.analyticsvidhya.com/blog/2021/05/multiclass-classification-using-svm/#:~:text=In%20its%20most%20basic%20type,which%20are%20binary%20classification%20problems..>
- [21] "flask library," [Online]. Available: [https://en.wikipedia.org/wiki/Flask_\(web_framework\)](https://en.wikipedia.org/wiki/Flask_(web_framework)).
- [22] [Online]. Available: https://en.wikipedia.org/wiki/Web_crawler.
- [23] "beautifulsoap library," [Online]. Available: [https://en.wikipedia.org/wiki/Beautiful_Soup_\(HTML_parser\)](https://en.wikipedia.org/wiki/Beautiful_Soup_(HTML_parser)).
- [24] "requestsLibrary," [Online]. Available: [https://en.wikipedia.org/wiki/Requests_\(software\)](https://en.wikipedia.org/wiki/Requests_(software)).