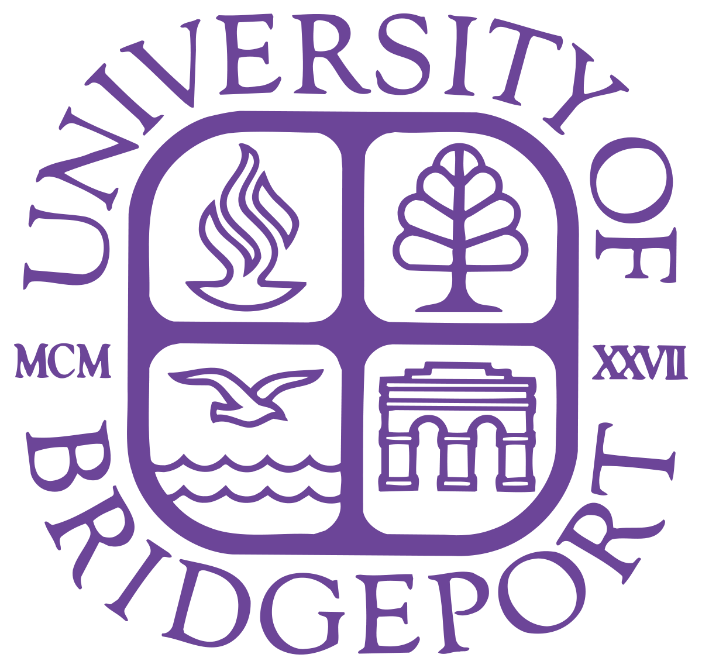
**WEB PAGE KEY-PHRASE EXTRACTOR**



*Project By:*

Srivatsan Ananthakrishnan (1021022)

*Under the Guidance of:*

Prof. Miad Faezipour

Department of Computer Science

University of Bridgeport

**Table of Contents**

[**Abstract**: 3](#_Toc512798046)

[Introduction: 3](#_Toc512798047)

[Use Cases 3](#_Toc512798050)

[**Algorithm:** 3](#_Toc512798051)

[Content Extraction 4](#_Toc512798052)

[Noun Chunker 4](#_Toc512798053)

[Ranking 4](#_Toc512798054)

[K-Means Grouping 4](#_Toc512798055)

[**Implementation and Steps:** 5](#_Toc512798051)

[Content Extraction 5](#_Toc512798052)

Requirements ………………………………………………………………………….......5

Data Set Configuration…………………………………………………………………… 5

Run Project ...………………………………………………………………………….......6

Notes……………………………………………………………………………………… 6

[HTML Parser… ...6](#_Toc512798052)

Installing BeautifulSoup………………………......…………………………………….....6

[Noun Extractor 6](#_Toc512798053)

Working …………………………………………………………………………...............7

[Ranking 7](#_Toc512798054)

[K-Means Grouping 7](#_Toc512798055)

Working …………………………………………………………………………...............8

[Tkinter for GUI 9](#_Toc512798055)

[**Screenshots**: 9](#_Toc512798057)

[**Conclusion and Future Work**: 1](#_Toc512798063)2

[**References**: 13](#_Toc512798063)

**ABSTRACT**

In this project we primarily consider the solution to the key phrase extraction problem based on Machine Learning and Keyword extraction algorithms. Since many of the key-phrase extraction has been implemented as extracting key-phrase from a document, database, news, article etc., whereas in the present paper, a new algorithm will be implemented with Machine learning technology and Python programming where a key-phrase extraction can be achieved dynamically from an active website, a large dataset and it can also be ranked with relevance scores of their occurrence. Another major problem in the traditional key-phrase extraction algorithm is the errors occurred with the extracted results due to non-meaningful phrases which can be solved using the recent algorithms developed by NLP technology which is also open source and would solve the mentioned problem.

The main goal of this project is to analyze any given input webpage and return the most relevant and related topics mentioned in the webpage. The system can be used in advertising fields. It helps in finding similarities between two webpages based on the relevant topics extracted. This analysis help advertising systems to push meaningful ads to a user, based on the user’s search history.

**INTRODUCTION**

Keyword-Key phrase extraction is an important task for summarizing a document or a webpage and giving top matched keywords. This project implementation processes a web page and returns a list of keyword phrases with relevance scores. It blends traditional natural language processing techniques with a machine learning ranking model applied to the web domain. For example, we can use a keyword extraction algorithm to tag and summarize a page with the most salient topics or to build a relationship graph between keywords. For example, by looking at which topics co-occur frequently with “event driven programming” we can find related topics.

There are many different approaches to topic extraction and summarization. In this project, a set of algorithms has been chosen to solve it. The major idea was taken from the Moz Developer Website [1].

**USE CASES**

Helps to tag and summarize a webpage.

It will help websites to push recommendations to a user.

**ALGORITHM**

1. Content Extraction / HTML PARSER
2. Noun Chunker: Identifying the nouns as candidates
3. Ranking: Ranking the candidates
4. K-means Grouping: Remove similar candidates to bring diversity to the candidates

**Content Extraction**

This algorithm is followed to extract the contents from a given input webpage.

The important parts of webpage which is of more importance is the title content, meta tag content, h1 tag contents, text part in the URL and the main text of the webpage must be extracted. All the other parts must be ignored.

**Noun Chunker**

The next step is to identify all the potential nouns in the extracted contents. All the potential nouns must be tagged based on the part of speech and adding it to the candidates list. From the candidates list, the potential noun phrases must be identified [2]. Limiting the candidates to noun phrases significantly reduces the set of candidates, while ensuring that the candidates are meaningful.

The second method to generate candidates looks up potential phrases in a modified version of Wikipedia article titles to find important entities that the noun-phrase chunker missed. For example, our noun-phrase chunker will split “Statue of Liberty” into two different candidates, but this step will add it as a candidate.

**Ranking the Candidates**

The candidates generated from the above step will now be ranked or given a score [1].

The ranking model is based on the following aspects

* Shallow: *Relative position in document, number of tokens, etc.*
* Occurrence: *Does the candidate occur in title, H1, meta description, etc.*
* Term frequency: *Count of occurrences, Average token count in the candidate, etc*.
* QDR: *Information retrieval motivated “Query-Document Relevance” ranking signals including TF-IDF (Term Frequency X Inverse Document Frequency), Probabilistic approaches, and Language models.* We used our open source library, QDR, to compute these scores [1].
* POS tags: *Is the keyword a proper noun, etc.?*
* URL features: *Does the keyword appear in the URL, etc.?*

**K-Means Grouping**

K-Means Clustering / Grouping is mainly used in the place where we deal with unlabeled data, i.e., data without defined categories or groups. The primary goal of this algorithm is to find groups in the data which are not explicitly labeled in the data [3]. Thus, once the algorithm runs and groups are identified, any new data can be easily assigned to the correct group.

The use of K-means in this project is to remove similar candidates to bring diversity to the candidates. Using K-Means cluster algorithm, we used to group similar words. This project is mainly based on the google-news data set and works well with news article web pages. This project uses google news pre-trained dataset which is used to group the similar words. The similar candidates will be grouped and the top keyword from each group will be displayed as an output.

**IMPLEMENTATION AND STEPS**

**Content Extraction**

The contents from the given URL is extracted using the third-party python package: newspaper article. By using this library, the URL title and contents are extracted and stored to a variable which is further used as an input for upcoming algorithms.

The Newspaper library, developed and maintained by Lucas Ou-Yang, is specially designed for extracting information from the websites of newspapers and magazines. The objective of this library is to extract and curate the articles from the newspapers and similar websites.

**Requirements:**

Version: Python 3.6

In Windows: Microsoft C++ 15 build tools

Link: <http://landinghub.visualstudio.com/visual-cpp-build-tools>

Commands to install libraries (tested in Windows 10):

*python -m pip install beautifulsoup4*

*python -m pip install newspaper3k*

*python -m pip install -U spacy*

*python -m spacy download en*

(It is okay if an error for shortcut link appears).

*python -m pip install wikipedia*

*python -m pip install numpy*

(If not there already)

*python -m pip install --upgrade gensim*

*python -m nltk.downloader punkt*

**Data set configuration:**

*GoogleNews-vectors-gensim-normed.bin* should be in same folder as python files for the application to run. It must be zipped along with other python source files.

**Run Project:**

*python project\_gui.py*

**Notes:**

Analyzer might take up to a few minutes to get the output based on the length of the article.This is because a lot of HTTP calls are occurring.

I have tested mainly the articles in *http://news.google.com*

Some of the test links are as follows:

*http://thehill.com/policy/cybersecurity/361583-4-legal-dimensions-of-the-uber-data-breach*

*http://news.bbc.co.uk/2/hi/health/2284783.stm*

*http://www.latimes.com/business/hiltzik/la-fi-hiltzik-net-neutrality-20171122-story.html*

**HTML Parser**

To scrap data from web pages, Python is extensively used. As the name suggests, HTML parser parses the web page’s HTML/XHTML contents and provides the information we are looking for [4] [5]. Thus, the HTML parsing is implemented with the help of **Beautiful Soup** Library in python.

**Beautiful Soup** is a [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) package for parsing [HTML](https://en.wikipedia.org/wiki/HTML) and [XML](https://en.wikipedia.org/wiki/XML) documents (including having malformed markup, i.e. non-closed tags, so named after [tag soup](https://en.wikipedia.org/wiki/Tag_soup)). It creates a parse tree for parsed pages that can be used to extract data from HTML, which is useful for [web scraping](https://en.wikipedia.org/wiki/Web_scraping) [6]. Parsing the URL through BeautifulSoup package and get its metatags, h1 tags and title/heading content [5]

**Installing Beautiful Soup:**

*$ apt-get install python-bs4 (for Python 2)*

*$ apt-get install python3-bs4 (for Python 3)*

*$ pip install beautifulsoup4*

Beautiful Soup is packaged as Python 2 code. When you install it for use with Python 3, it’s automatically converted to Python 3 code. If you don’t install the package, the code won’t be converted. There have also been reports on Windows machines of the wrong version being installed [5].

**Noun Extractor**

The next phase in Key-Phrase analyzer is to extract the potential noun which is going to be the candidate. For example, extracting single word nouns from text content and searching related phrases from wikipedia articles [1]. In this project, **SpaCy** library in python is used to find out the noun in the resultant title and the content.

How Candidates are chosen with this algorithm?

* If word is a proper noun and not too short word length.
* If word is a noun that is not too short and not belonging to WH words like who, what, where etc.,
* Date nouns are excluded.
* Proper nouns are excluded that doesn’t have a definite entity type.

**Working:**

The Noun Extraction is implemented in this project mainly using two python libraries namely, **SpaCy** and **Wikipedia** library.

**Wikipedia** is a Python library that makes it easy to access and parse data from Wikipedia.

**SpaCy** is another python library that is used for natural language processing and for part of speech tagging. SpaCy can parse and tag all the words in each document/set of words. Thus, by using this feature, we get all the nouns and proper nouns.

Wikipedia library to identify any potential phrases in the content. This is achieved by extracting nouns from the content of the article and get related keywords from wikipedia [7]. SpaCy library identifies nouns from the content and for each identified noun word, we do a search in wikipedia to get 50 search results.

If any of the search result is in article content, that phrase is taken as a potential keyword along with the noun.

**Ranking**

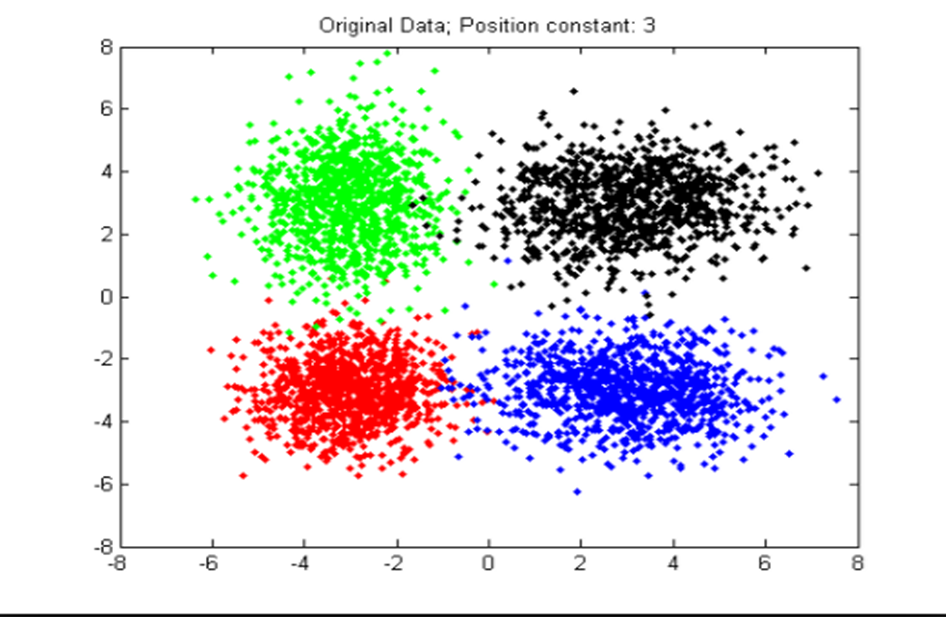
The next step is to rank the words and phrases (tokens)from the above result based on factors like occurrence, frequency etc.,

Ranking the words based on where the word has occurred like title, URL, H1, meta content etc., and if it is a phrase, Number of occurrences, number of words in the token.

This ranking methodology is designed by me based on the requirements of this project.

**K-Means Grouping**

Clustering is the process of grouping a set of physical or abstract objects into classes of similar objects. A cluster is a collection of data objects that are like one another within the same cluster and are dissimilar to the objects in other clusters [8]. The below figure shows the graphical representation of the K-Means clustering that groups the keywords and key-phrases of similar context into same cluster which is been represented in different colored dots.



*Figure 1: K-Means Clustering*

In this project, to remove keywords that are similar, we used the concept of clustering / grouping. This is to bring more diversity to the top ranked keywords.

**Working:**

We use an algorithm called K-means for grouping. The K-means is one of algorithm that commonly used in clustering process is K-means clustering. The “K” in its name refers to the fact that the algorithm looks for a fixed number of clusters which are defined in terms of proximity of data points to each other [8]. There is no use of any other third-party algorithm other than K-Means clustering is used for grouping.

Other than *numpy* (for computation) and *gensim* (loading of data set), each word in the keyword list is converted to a vector using *word2vec*.

We use GoogleNews pre-trained dataset to get vector of word [9]. Then, we divide the n keywords into k clusters. Initially, we randomly choose k centroids. The Euclidian distance of each vector is calculated from the centroid [10]. Each cluster is formed with a centroid and vectors which are closest to it. Once cluster is formed, a new centroid is found for each cluster [11].

Based on the distance, new clusters are formed. This process is continued until, the list of centroid stops changing. This is the final list of clusters. Only one keyword from each cluster will be taken as final keyword.

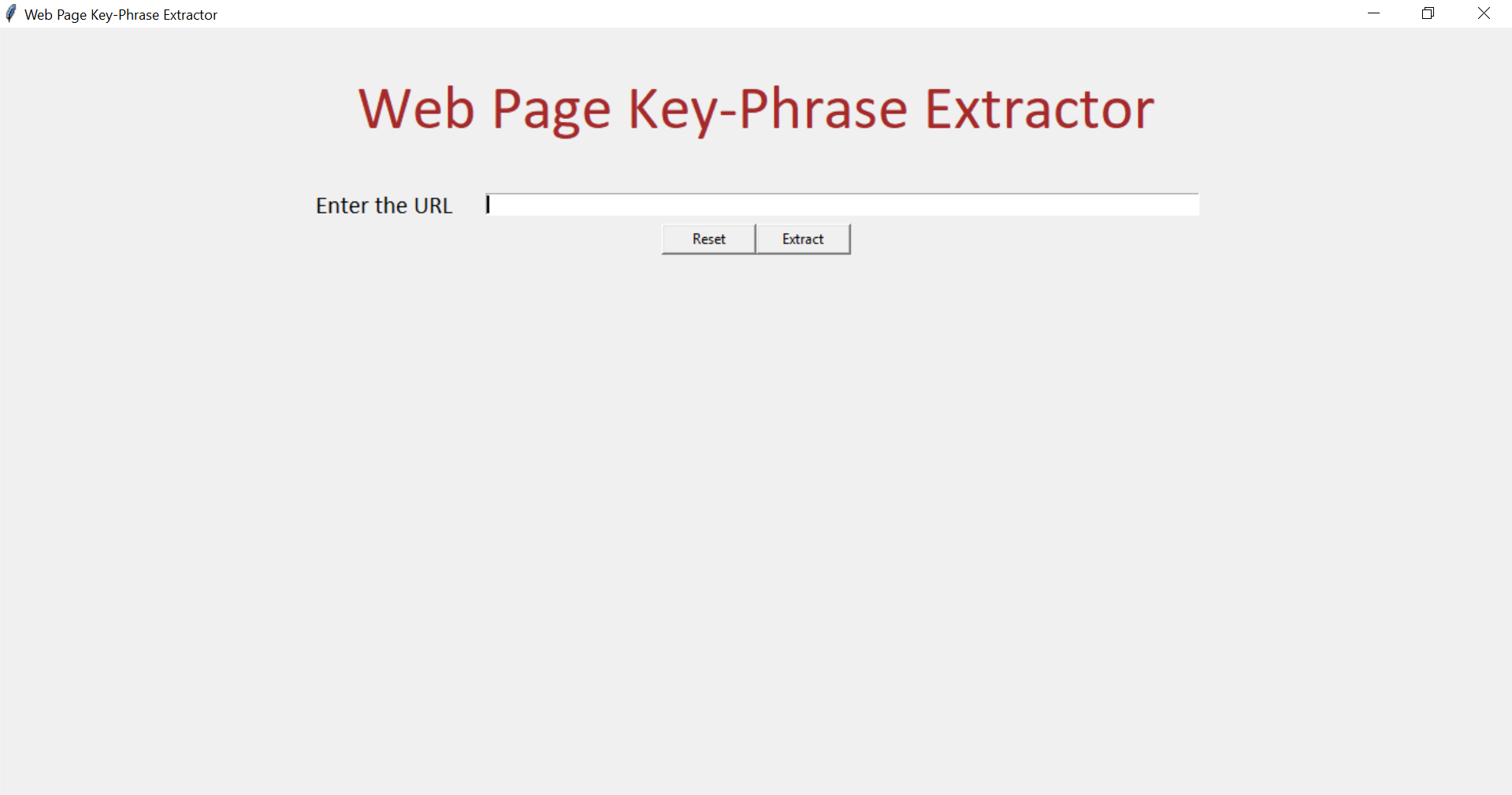
Once the Grouping of the words is completed, the next process is to sort all the words and phrases based on the ranking scores and displaying Top 15 words.

**Tkinter for GUI**

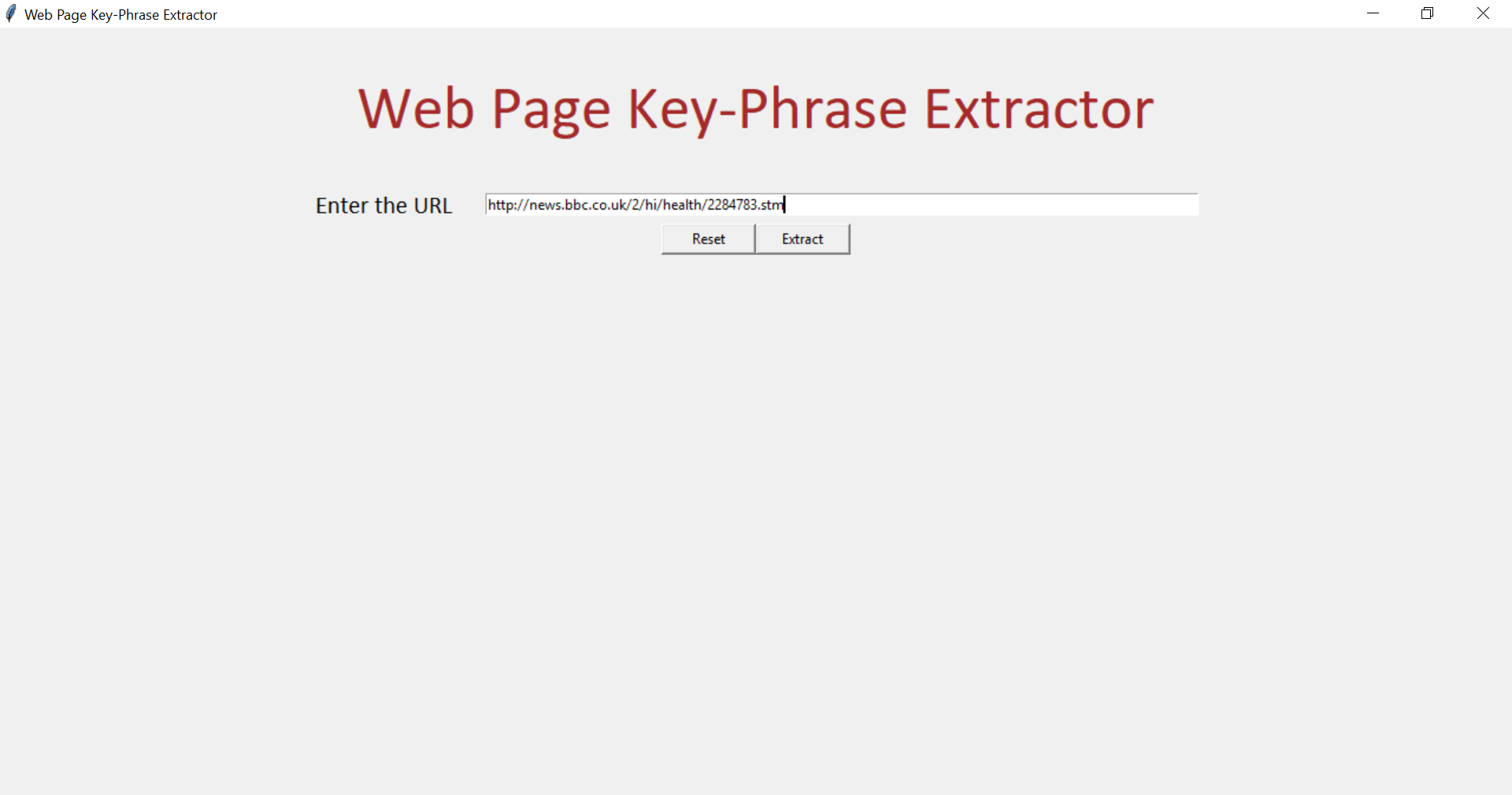
The [**Tkinter**](https://docs.python.org/2/library/tkinter.html#module-Tkinter) module (“Tk interface”) is the standard Python interface to the Tk GUI toolkit. Our project has a basic website that gets the news or any article from website as an input and process all the algorithms internally and finally displays the most important key-phrases as the desired output. Tkinter package is used in our code to implement the webpage.

**SCREENSHOTS:**

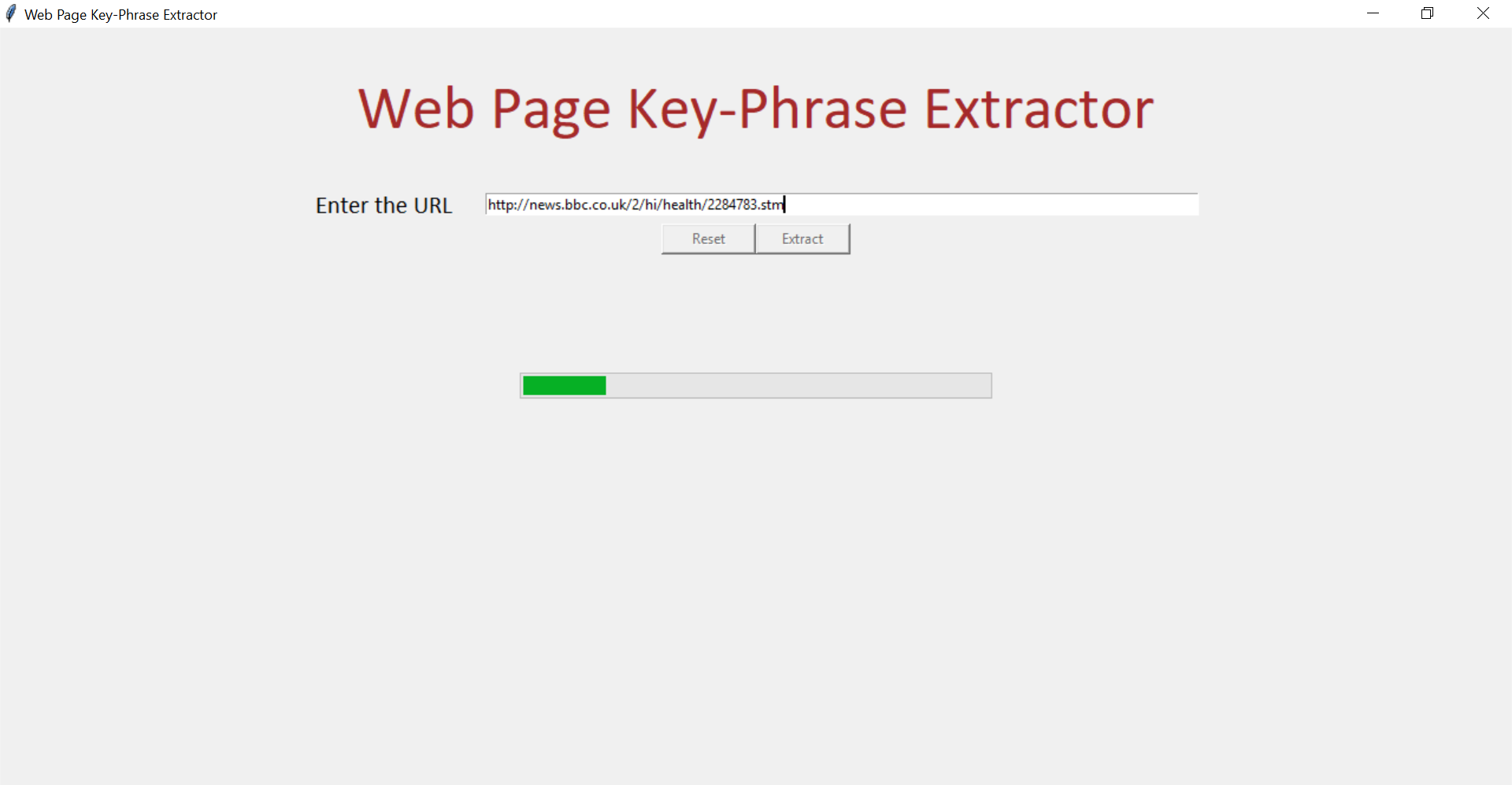
* Enter the URL to extract the key-phrase from the website.



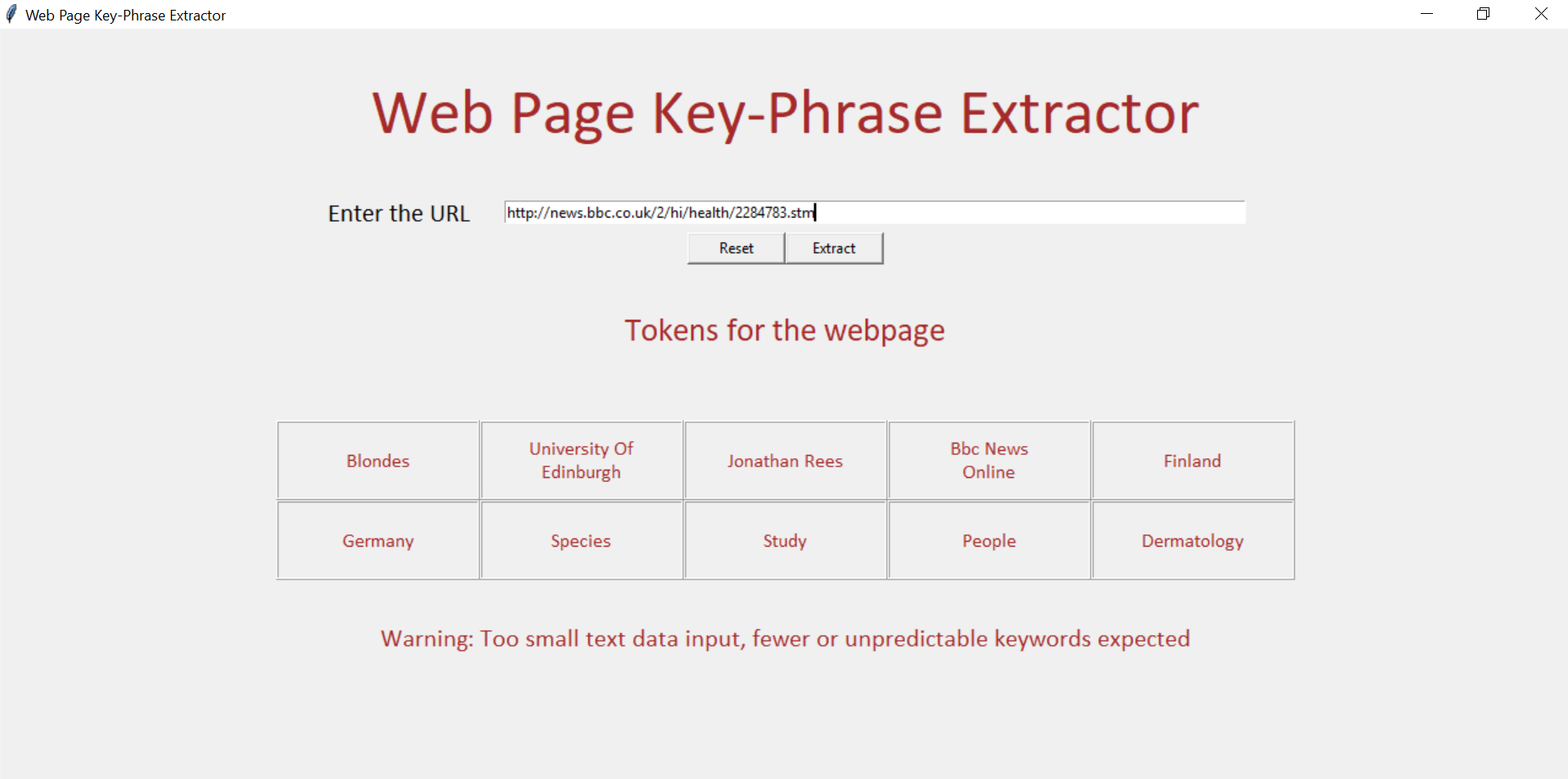
* Valid URL is entered



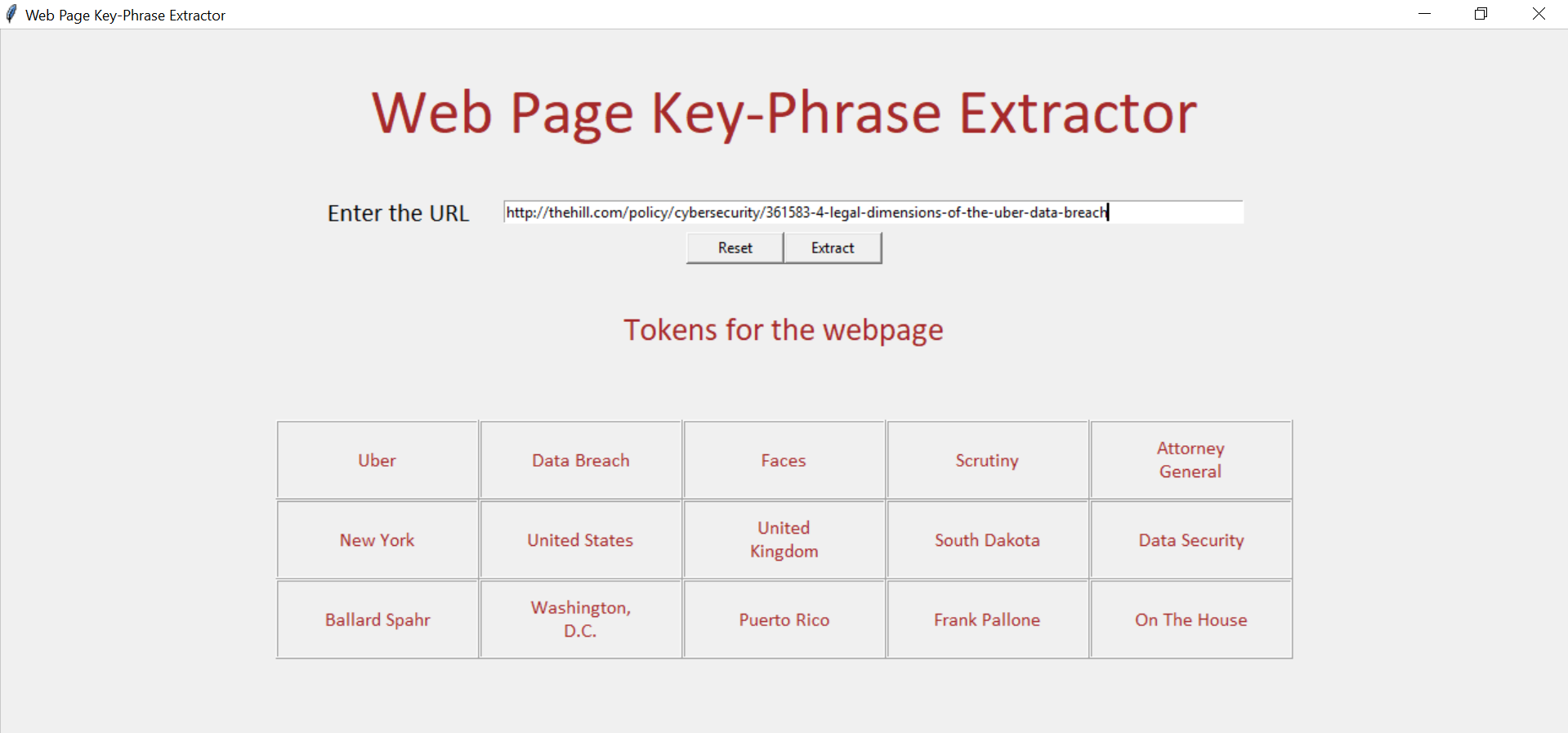
* Extracting the potential keyword, key-phrase from the entered website



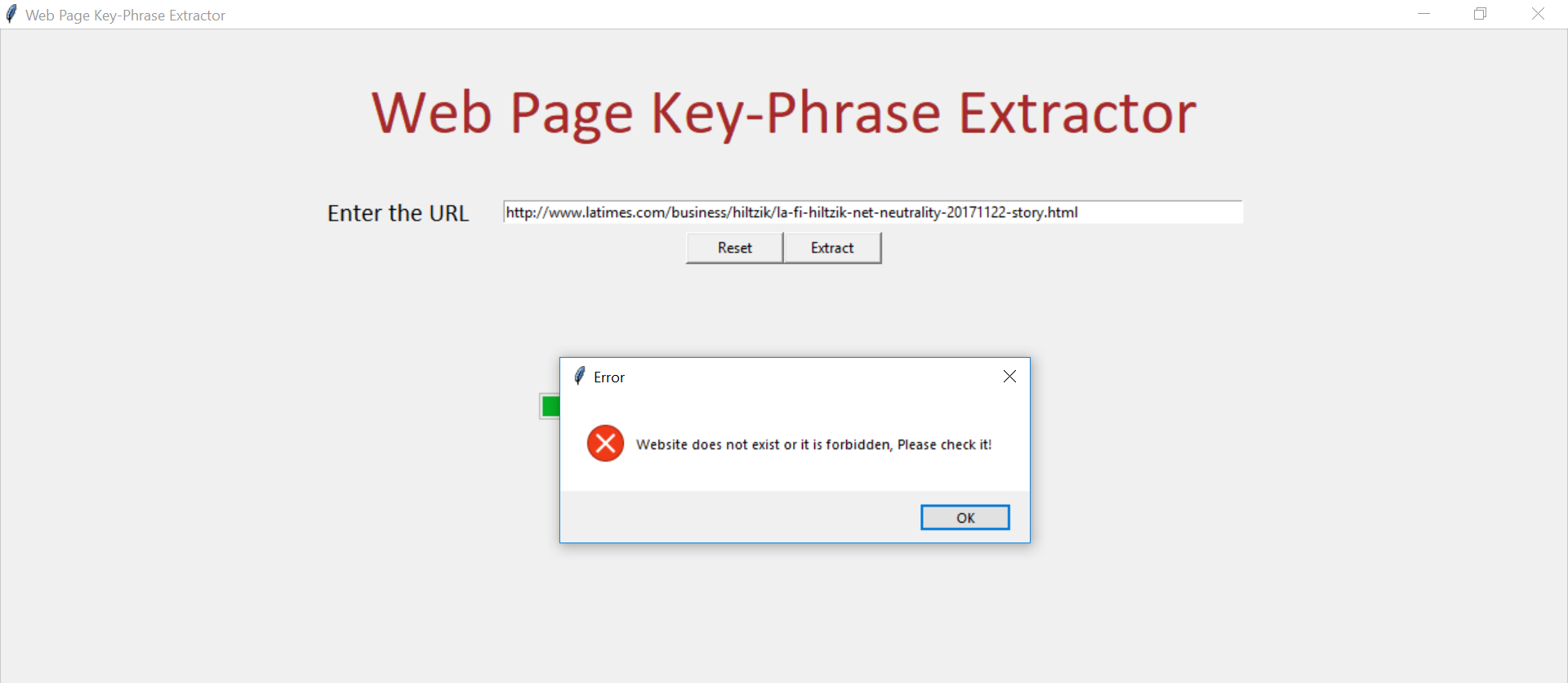
* Case1: Too small extracted keywords from the website.

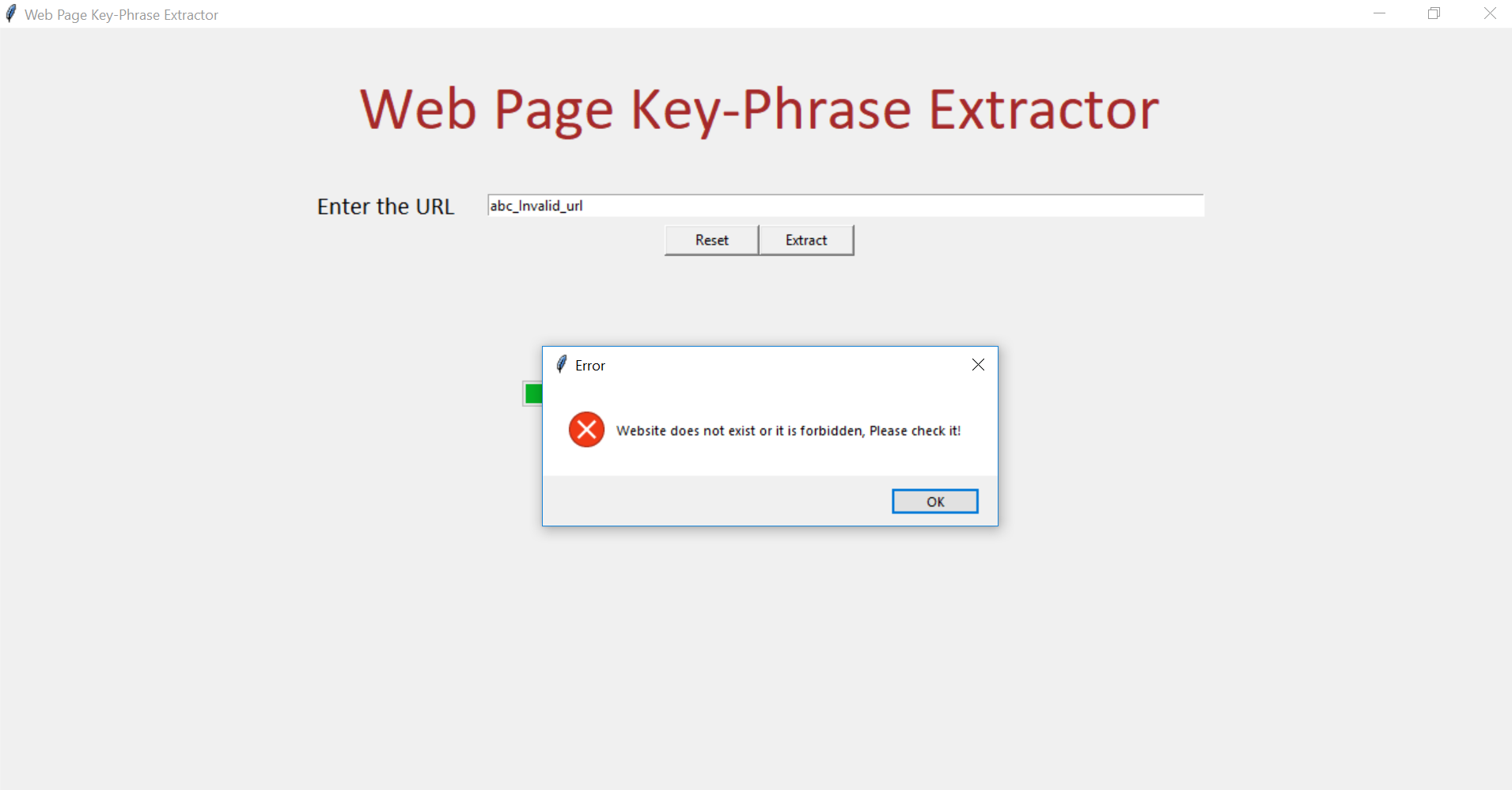


* Case2: Results with 15 potential keyword phrases from the website.

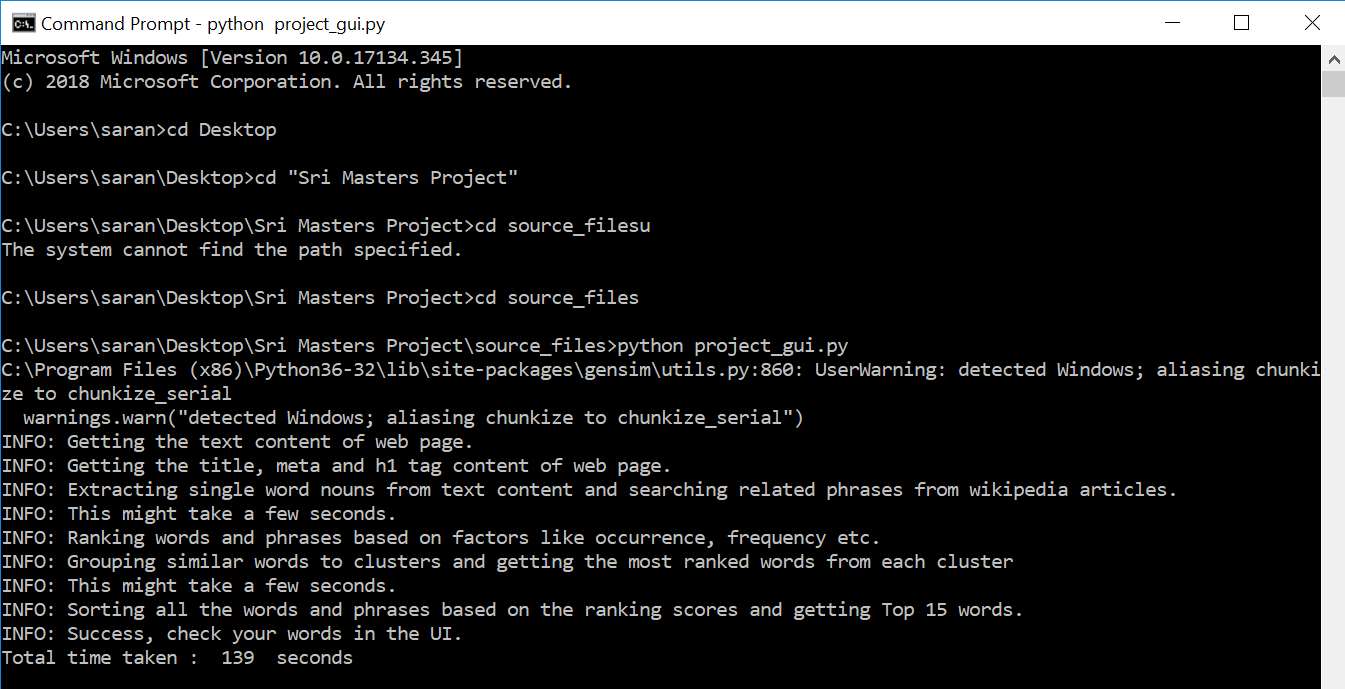


* Case 3: Invalid webpage, URL.





* The process execution snapshot from command prompt.



**Conclusion and Future Work:**

This project was mainly developed by using third party algorithms and python libraries. Most of the errors and difficulties faced during the implementation is when designing the Noun Chunker phase where it ends up in extracting few words that are not meaningful and doesn’t form a phrase. Thus, I have implemented the noun chunker with SpaCy open source parser which is most effective in extracting the nouns. As a part of future work, this project is mainly developed for news articles and magazines and the google pre-trained dataset is used which is used to group similar words in K-means clustering. Thus, as a future development, this project can be widely used by all webpage contents as a push recommendation to user on a large scale and the running time can be further optimized by using multi-threading or any advanced optimization techniques.

# **References**

|  |  |
| --- | --- |
| [1] | M. Peters, "Moz Developer Blog," 25 aug 2016. [Online]. Available: https://moz.com/devblog/machine-learning-approach-to-keyword-extraction/. |
| [2] | K. S. a. V. N. Hasan, "Automatic Keyphrase Extraction: A Survey of the State of the Art," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2014. |
| [3] | A. Trevino, "Introduction to K-means Clustering," 6 Dec 2016. [Online]. Available: https://www.datascience.com/blog/k-means-clustering. |
| [4] | [Online]. Available: https://www.pythoncentral.io/html-parser/. |
| [5] | "Beautiful Soup Documentation," [Online]. Available: https://www.crummy.com/software/BeautifulSoup/bs4/doc/. |
| [6] | "Web Scraping with BeautifulSoup," 09 march 2016. [Online]. Available: http://www.pythonforbeginners.com/python-on-the-web/web-scraping-with-beautifulsoup/. |
| [7] | "Deciding on an Article Title," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Wikipedia:Article\_titles#Deciding\_on\_an\_article\_title. |
| [8] | K. Kusrini, "Grouping of Retail Items by Using K-Means Clustering," *The Third Information Systems International Conference,* 2015. |
| [9] | "SpaCy: how to load Google news word2vec vectors?," Stack Overflow, [Online]. Available: https://stackoverflow.com/questions/42094180/spacy-how-to-load-google-news-word2vec-vectors. |
| [10] | Y. Gu, "Different ways to calculate the euclidean distance in python," Code Hamster, 09 march 2018. [Online]. Available: http://www.codehamster.com/2015/03/09/different-ways-to-calculate-the-euclidean-distance-in-python/. |
| [11] | Python, "Python K-Means Data Clustering and finding of the best K," 22 June 2017. [Online]. Available: https://learn.scientificprogramming.io/python-k-means-data-clustering-and-finding-of-the-best-k-485f66297c06. |