**WEB SCRAPPING TUTORIAL FOR VIKASPEDIA**

**URL** - <https://vikaspedia.in/agriculture>

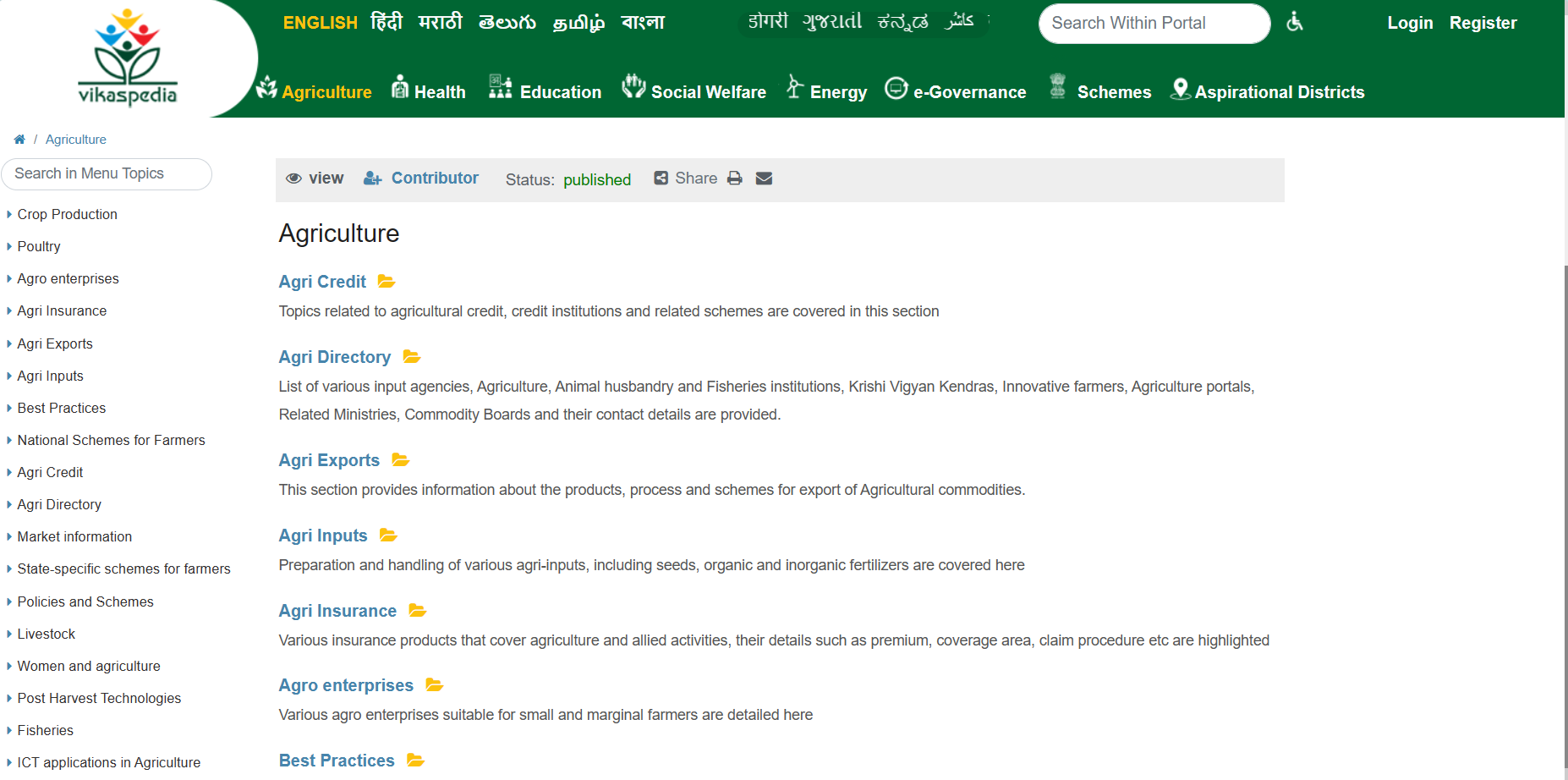
**REQUIREMENTS –**

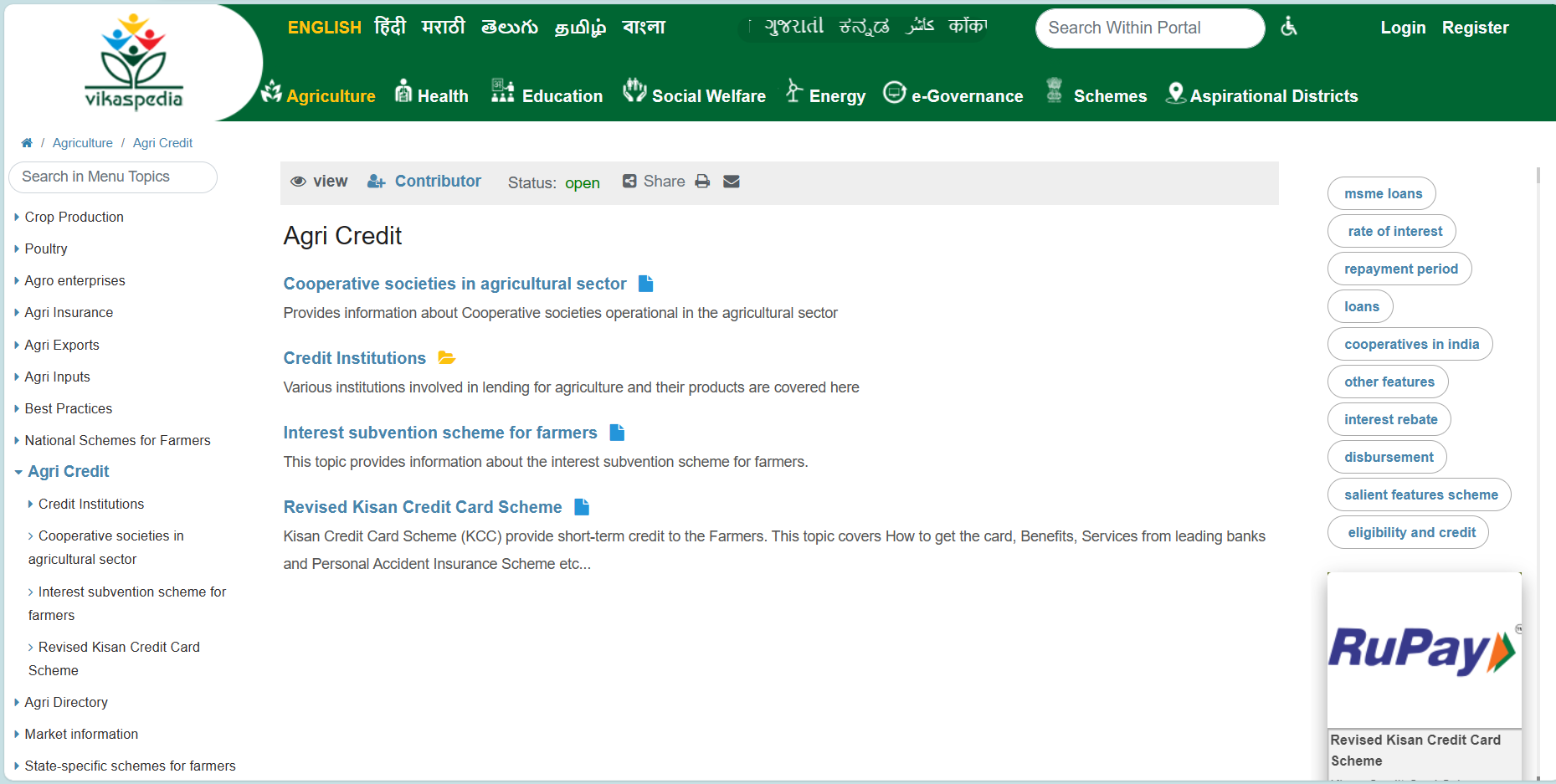
**pip install requests beautifulsoup4**

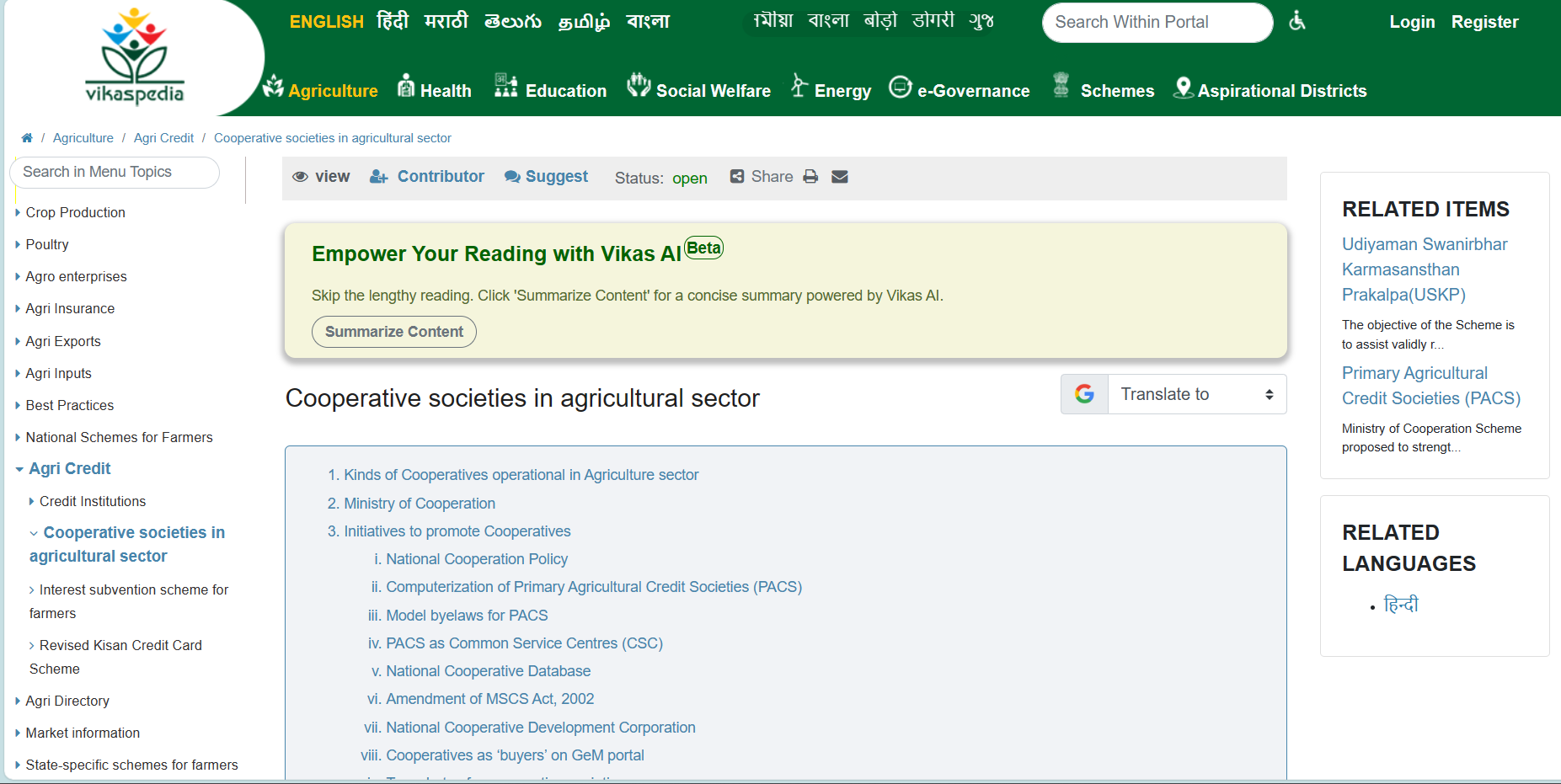
|  |  |  |
| --- | --- | --- |
| **Sr. no.** | **NAME** | **DESCRIPTION** |
| 1 | requests | Python library for making HTTP requests to web pages. |
| 2 | bs4 | BeautifulSoup library for parsing HTML. |
| 3 | time | Standard Python library for time-related functions. |
| 4 | os | Standard Python library for operating system dependent functionality (used for interacting with the file system). |
| 5 | re | Standard Python library for regular expressions, used for text processing |
| 6 | html.parser | A parser specified in BeautifulSoup to handle HTML.  In this case, it refers to the parser provided by the Python standard library. |

**USAGE** -

First Look of the website







By this we can understand, the structure of the website.

Through this we understood

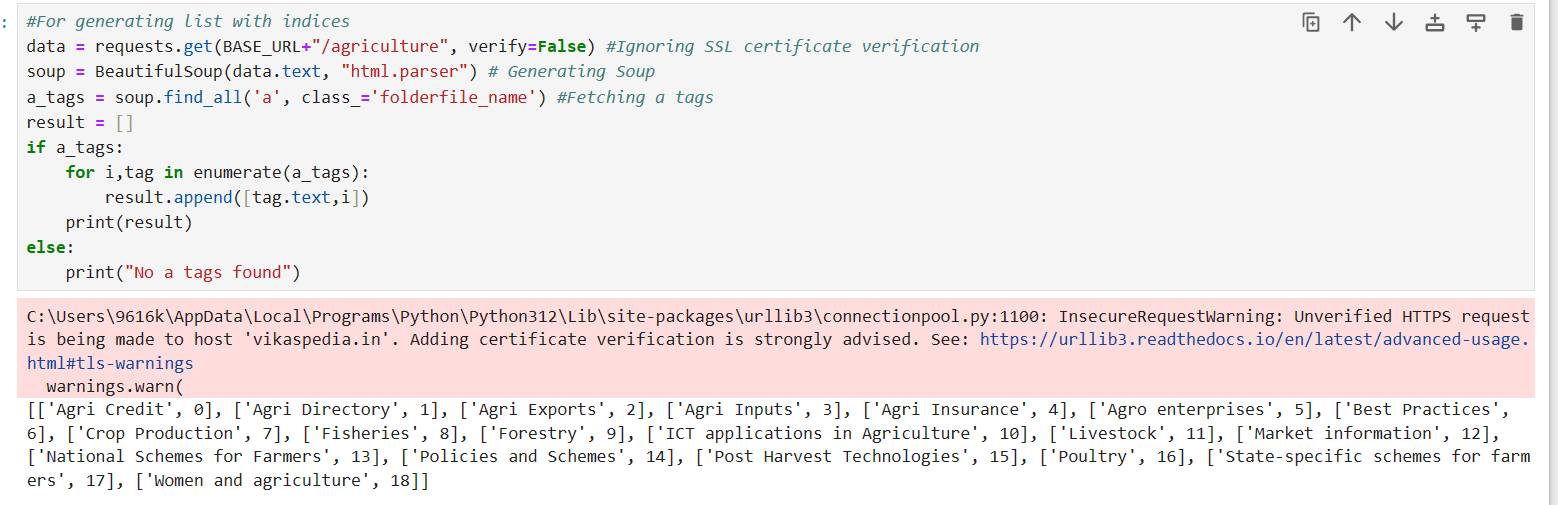
* Folders and Direct documents are assigned class “folder”
* If no a tags found with that class, we have to scrap content, in Fig 3

If we have a folders, we expand and scrap content

**HOW TO USE?**

1. First check how many a\_tags are there in 1st page/ homepage

In our case we had 19

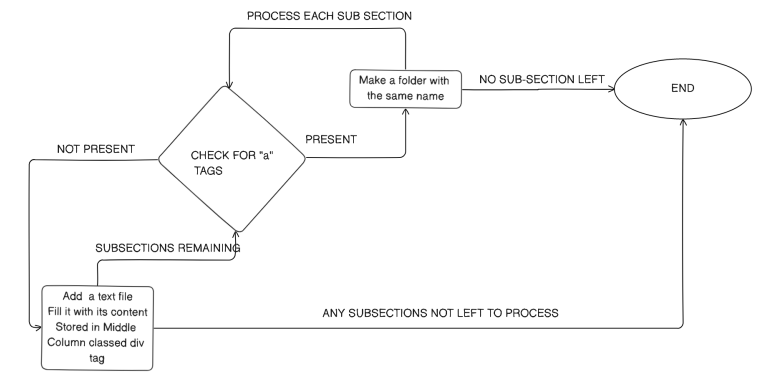


1. Set your parameters, how much of data to be extracted

*RESULT\_VARIABLE=webScrape(URL,START,END,RESULTING\_DIRECTORY)*

*eg- allSectors=webScrape(BASE\_URL+"/agriculture",16,18,"VIKASPEDIA\_DATA")*

*#START AND END ARE BOTH INCLUDING INDICES*



**WebScrape Function:**

Start

|

|--> Wait for 10 seconds

|

|--> Send HTTP GET request to the specified URL (ignoring SSL certificate verification)

| |

| |--> Successfully received response?

| | |

| | |--> Parse HTML content using BeautifulSoup

| | |

| | |--> Find all anchor tags with class 'folderfile\_name'

| | |

| | |--> Any anchor tags found?

| | | |

| | | |--> Iterate through selected anchor tags (from start index to end+1)

| | | | |

| | | | |--> Clean category name and calculate category folder path

| | | | |

| | | | |--> Create category folder if it does not exist

| | | | |

| | | | |--> Recursively call webScrapeSubCategory for subcategories

| | | | |

| | | | |--> Append category and subcategory data to the result list

| | | |

| | | |--> No anchor tags found

| | | | |

| | | | |--> Scrape title and main content of the page

| | | | |

| | | | |--> Write scraped data to a text file in the specified folder path

| | |

| | |--> Any exceptions occurred during scraping?

| | | |

| | | |--> Handle exceptions and print an error message

|

|--> End (Return result list containing scraped data)

**webScrapeSubCategory**

This function is designed to scrape data from a specified URL without any limitations on the number of anchor tags. It is same as the webScrape function above

1. After making Folders, extract data from a single page, by function fetchData

Start

|

|--> Send HTTP GET request to the specified URL (ignoring SSL certificate verification)

| |

| |--> Successfully received response?

| | |

| | |--> Parse HTML content using BeautifulSoup

| | |

| | |--> Find the <div> element with id='MiddleColumn\_internal'

| | |

| | |--> Remove all <a> tags from MiddleColumn

| | | |

| | | |--> Iterate through all <a> tags inside MiddleColumn

| | | | |

| | | | |--> Remove the <a> tag from the MiddleColumn contents

| | |

| | |--> Extract text content from specific HTML elements (h3, h4, p)

| | | |

| | | |--> Iterate through MiddleColumn contents

| | | | |

| | | | |--> Check if the element is <h3>, <h4>, or <p>

| | | | | |

| | | | | |--> Add element's text content to text\_content variable

| | | | | |

| | | | |--> If the element is not <h3>, <h4>, or <p>

| | | | | |

| | | | | |--> Add element as it is to text\_content variable

| | |

| | |--> Remove HTML tags from text content using regular expressions

| |

| |--> End (Return cleaned and formatted text content)

|

|--> End