

WEB DATA EXTRACTION FROM WEBSITES

K.VENKATA ADITHYA

LINKEDIN : www.linkedin.com/in/venkataadithyakolluru

GITHUB : www.github.com/VenkataAdithyaKolluru

CONTENTS

- Abstract
- Introduction
- Existing System
- Drawbacks
- Proposed System
- Objectives
- Advantages
- Architecture
- Module Diagrams
- Implementation
- Result
- Conclusion
- References



ABSTRACT

Web scraping, a dynamic technique, has emerged as an indispensable tool for gathering structured information from websites and online sources. Web scraping encompasses a wide range of automated data extraction methods that enable users to access and collect data from websites and web applications. These techniques involve parsing the HTML, XML, or other structured data formats of web pages to extract specific information, such as text, images, links, and more. Web scraping plays a pivotal role in the digital age, offering unprecedented access to valuable data for research, business, and decision-making purposes. As the digital frontier continues to expand, web scraping remains a powerful tool for unlocking insights and opportunities from the vast online ecosystem. Web scraping should be conducted ethically and within legal boundaries. Some websites explicitly prohibit web scraping in their terms of service and scraping sensitive or personal data without consent is illegal in many jurisdictions. Web scraping is a powerful tool for collecting and utilizing data from the web.


INTRODUCTION

WEB DATA EXTRACTION FROM WEBSITE


Web scraping is a technique to fetch data from websites. While surfing on the web, many websites don't allow the user to save data for personal use. One way is to manually copy-paste the data, which is both tedious and time-consuming. Web Scraping is the automation of the data extraction process from websites. This event is done with the help of web scraping software known as web scrapers. They automatically load and extract data from the websites based on user requirements. These can be custom-built to work for one site or can be configured to work with any website.

Existing System


1. Bright Data:

Bright Data provides fully compliant and risk-free access to robust data. With its customizable dashboard and the capability to structure data sets of any size 

2. Scrapingdog:

Designed to lend easy web scraping to developers and non-developers alike 

3. AvesAPI:

AvesAPI provides a highly focused extraction of structured data from Google Search 

Drawbacks

- **Legal and Ethical Issues:** Scraping data can infringe on copyrights and terms of service, leading to legal consequences.
- **Data Quality and Reliability:** Scraped data may be unstructured, inaccurate, or outdated, requiring significant cleaning and verification.
- **Technical Challenges:** Websites often implement anti-scraping measures, which can make scraping difficult or impossible without advanced techniques.

Proposed System

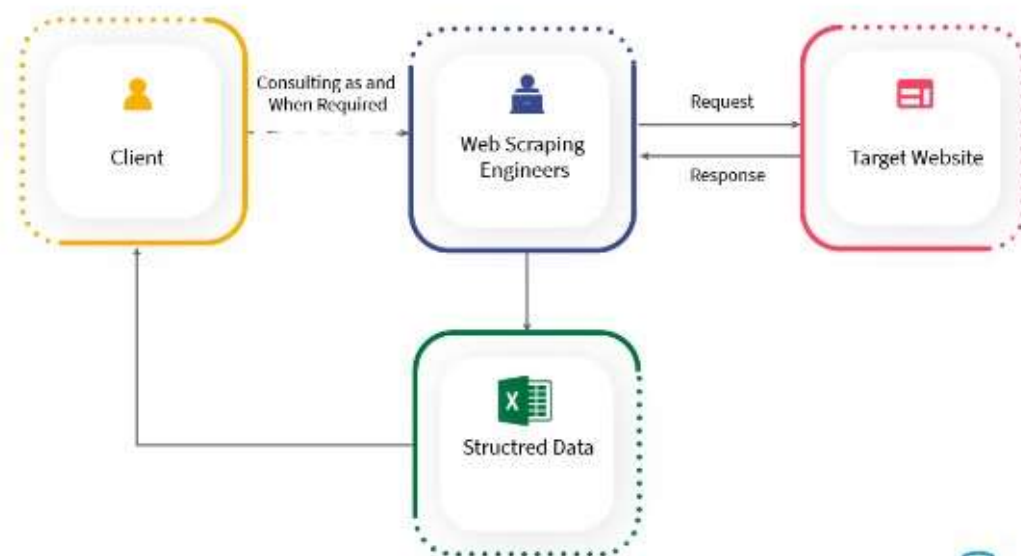
Data Collection

Data Processing

Target Websites

Scalability Error

Handling and Logging



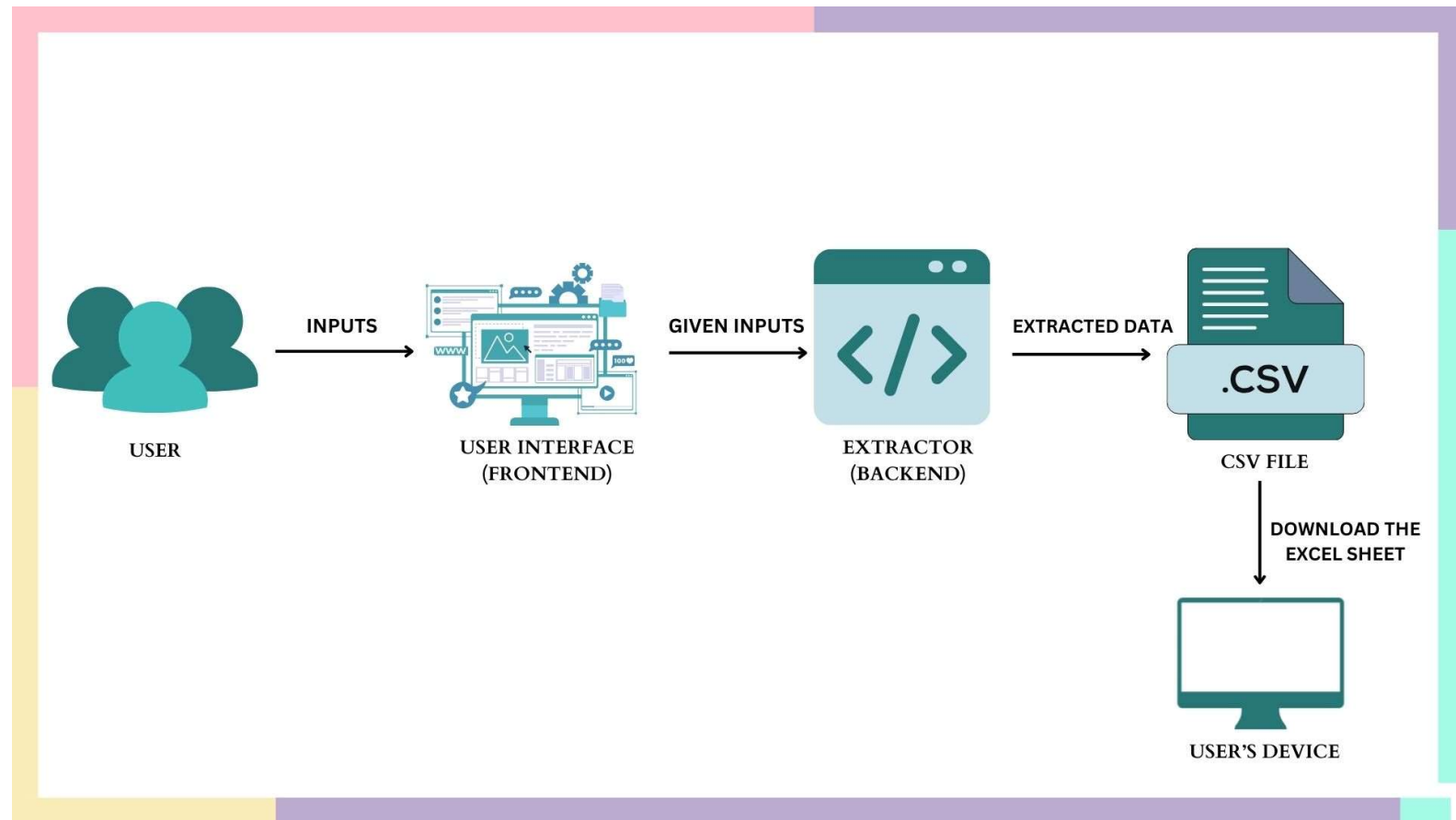
Objectives

- **Monitoring price** and its fluctuations in set fields like real estate and getting that data.
- **Extract news and articles** from various sites and efficiently, gather news content for analysis or reading and Monitor multiple sources simultaneously.
- **Monitor changes** on a specific website and track product releases and news. Receive timely notifications about relevant website modifications. Keep up-to-date without manual checking.
- **Obtaining large datasets** scraping allows organizations to collect vast amounts of data from websites, which can be used for various purposes such as market research, analysis, and decision-making.
- **Extract information from public records** available on websites to gather data related to individuals, businesses, or government activities .Streamline the process of obtaining public information.

Advantages

- Achieve Automation
- Business Intelligence & Insights
- Unique and Rich Datasets
- Create Applications Without Public APIs

Architecture



Modules

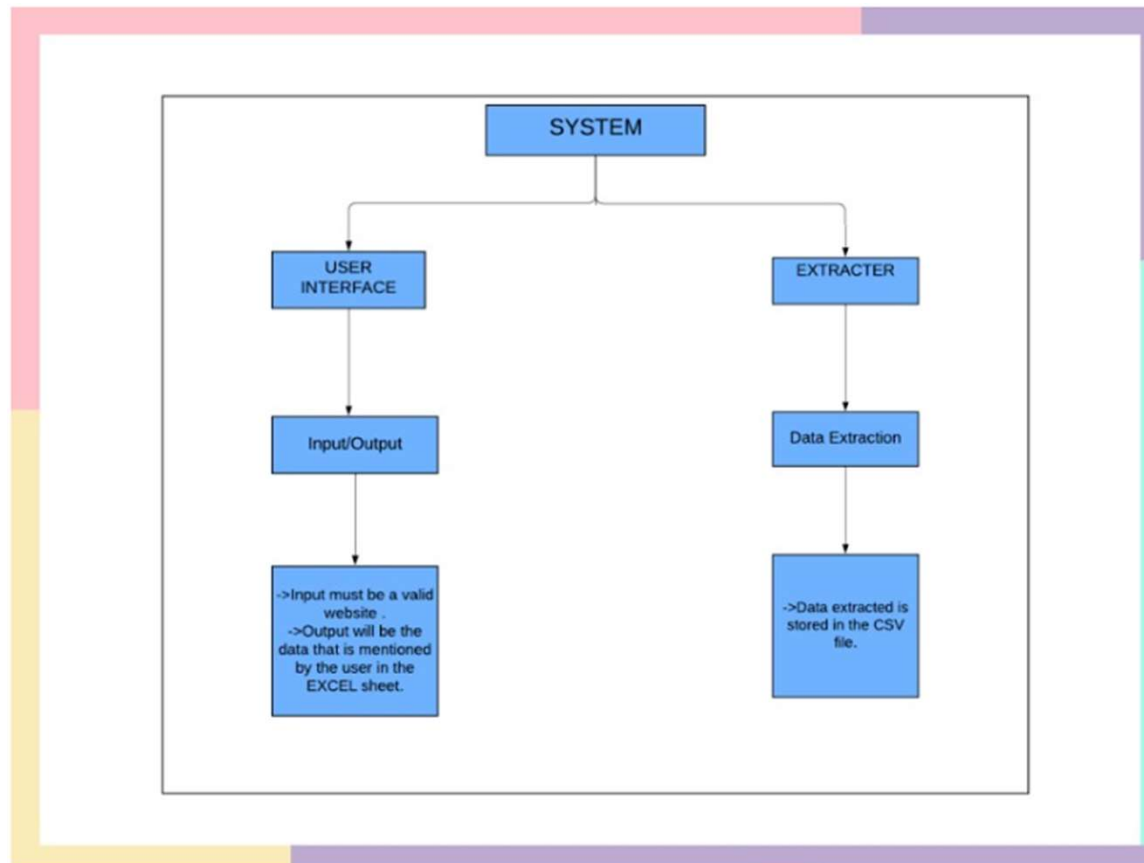
The proposed system consists of two modules as mentioned in the above diagram.

- User Interface
- Extractor

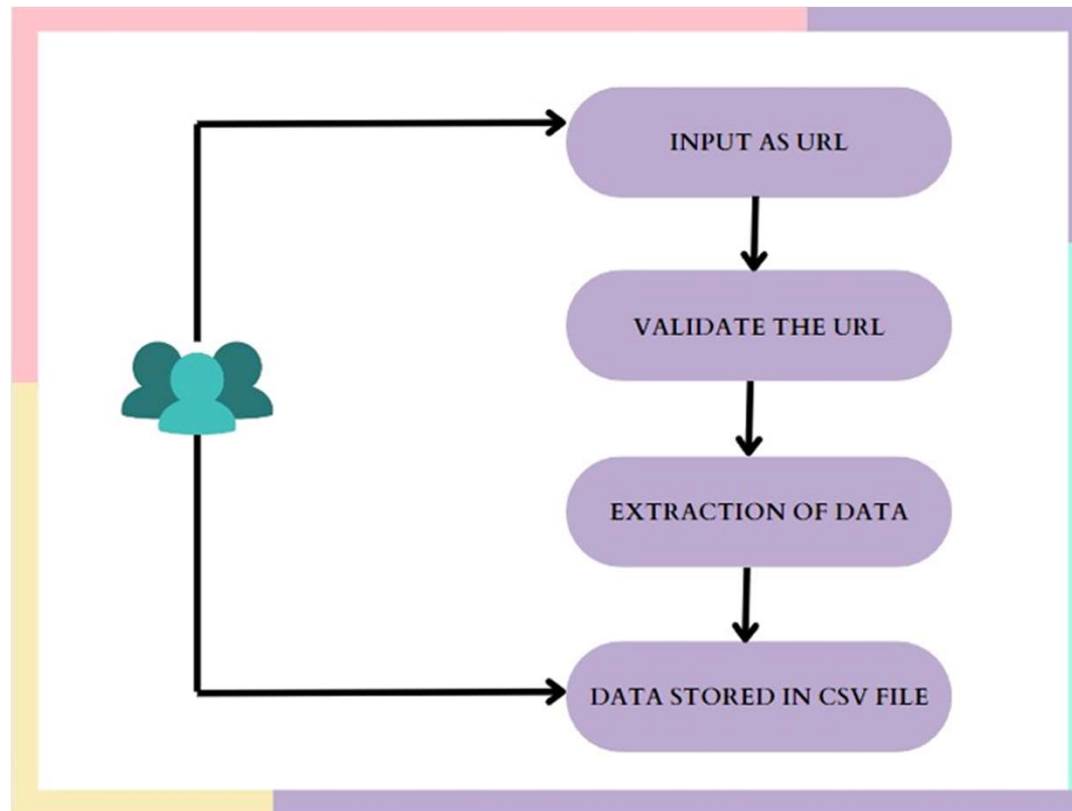
USER INTERFACE: The functions in the user interface module include users giving the required inputs. Once the user gives the required input, the inputs are passed to the extractor to begin the extraction. This module allows the users to give input and then as output the excel file containing the extracted data is downloaded into the user's device.

EXTRACTOR: The csv modules is used to store the csv file temporarily which contains the required data and downloads that csv file as an excel sheet into the user's device.

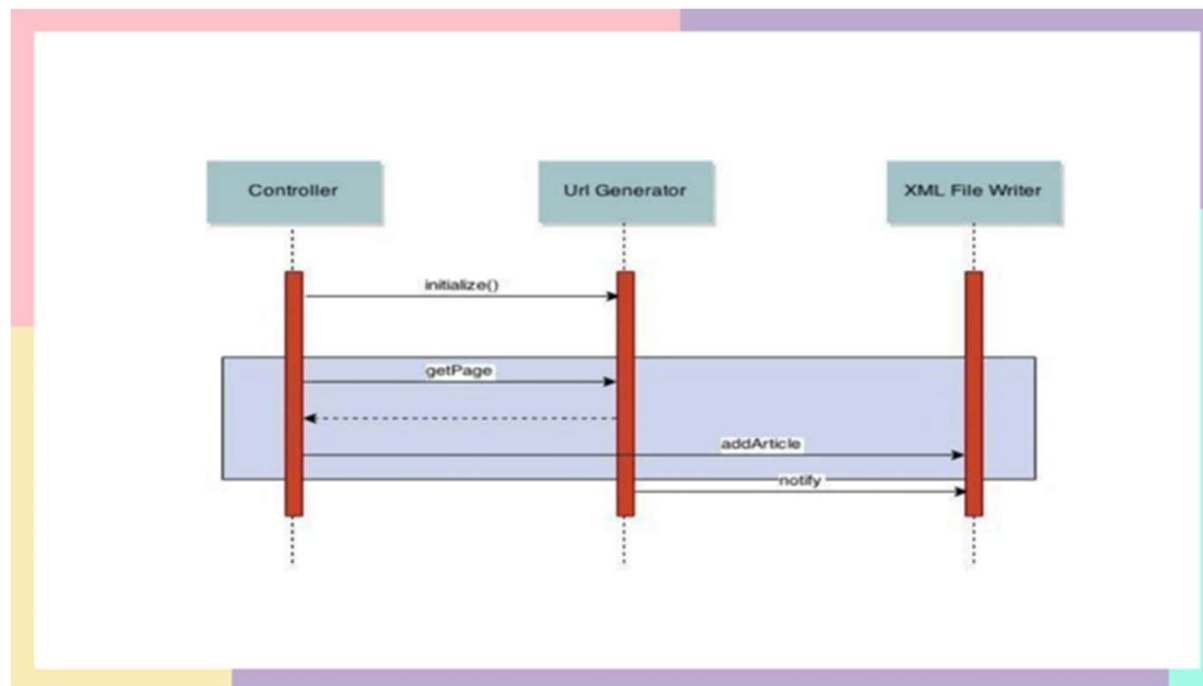
Modules



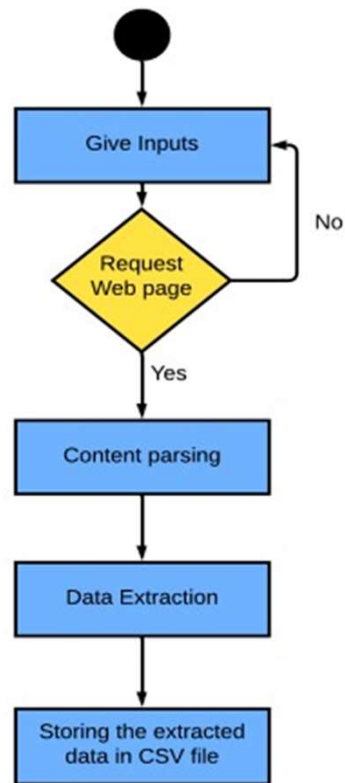
Use Case Diagram



Sequence Diagram



Activity Diagram



Implementation

PYTHON SOURCE CODE : app.py

```
from flask import Flask, request, render_template, send_file
import pandas as pd
import requests
from bs4 import BeautifulSoup

app = Flask(__name__)

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/scrape', methods=['POST'])
def scrape():
    url = request.form['url']
    tag = request.form['tag']
    class_name = request.form['class_name']
    file_name = request.form['file_name']

    response = requests.get(url)
    soup = BeautifulSoup(response.content, 'html.parser')

    if class_name:
        data = soup.find_all(tag, class_=class_name)
    else:
        data = soup.find_all(tag)

    extracted_data = [item.get_text(strip=True) for item in data]

    df = pd.DataFrame(extracted_data, columns=['Data'])
    csv_path = f"{file_name}.csv"
    df.to_csv(csv_path, index=False)
```

```
        return send_file(csv_path, as_attachment=True)

if __name__ == '__main__':
    app.run(host='127.0.0.1', port=5000, debug=True)
```

Implementation

- index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Web Scraper</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <div class="container">
    <h1>WEB SCRAPING FROM WEBSITES</h1>
    <form action="/scrape" method="post">
      <label for="url">Website URL:</label>
      <input type="text" id="url" name="url" required>
      <label for="tag">HTML Tag:</label>
      <input type="text" id="tag" name="tag" required>
      <label for="class_name">Class Name (optional):</label>
      <input type="text" id="class_name" name="class_name">
      <label for="file_name">CSV File Name:</label>
      <input type="text" id="file_name" name="file_name" required>
      <input type="submit" value="SCRAPE">
    </form>
```

```
</div>
<div class="bottom-right-text">
  K.VENKATA ADITHYA - 22R21A6792 (Team Lead)
  <br>
  K.TAUFEEQ UMAR - 22R21A6786
  <br>
  B.VARUN - 22R21A6768
</div>
</body>
</html>
```

Implementation

- style.css

```
body {
  font-family: Arial, sans-serif;
  background-color: #80CBC4;
  margin: 0;
  padding: 0;
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100vh;
}

.container {
  background-color: #546E7A;
  padding: 2em;
  border-radius: 8px;
  box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
  max-width: 500px;
  width: 100%;
  margin-bottom: 1em;
  border: 3px solid #000000;
}

h1 {
  margin-bottom: 1em;
  font-size: 2em;
  color: #00BCD4;
  text-align: center;
```

```

}

form {
  display: flex;
  flex-direction: column;
}

label {
  margin-bottom: 0.5em;
  color: #ffffff;
}

input[type="text"] {
  padding: 0.5em;
  margin-bottom: 1em;
  border: 3px solid #000000;
  border-radius: 4px;
  font-size: 1em;
}

input[type="submit"] {
  padding: 0.5em;
  background-color: #00BCD4;
  color: #0e5345;
  font-weight: bold;
  border-radius: 15px;
  font-size: 1em;
  cursor: pointer;
  width: 100px;
  margin-left: 200px;
  height: 45px;
}

}
```

VENKATA ADITHYA KOLLURU

```
input[type="submit"]:hover {
  background-color: #179f84;
}

.bottom-right-text {
  position: absolute;
  bottom: 20px;
  right: 20px;
  background-color: #117864;
  color: #fff;
  padding: 10px;
  border-radius: 10px;
  font-size: 0.9em;
  margin-bottom: 1em;
  border: 3px solid #000000;
}
```


Results

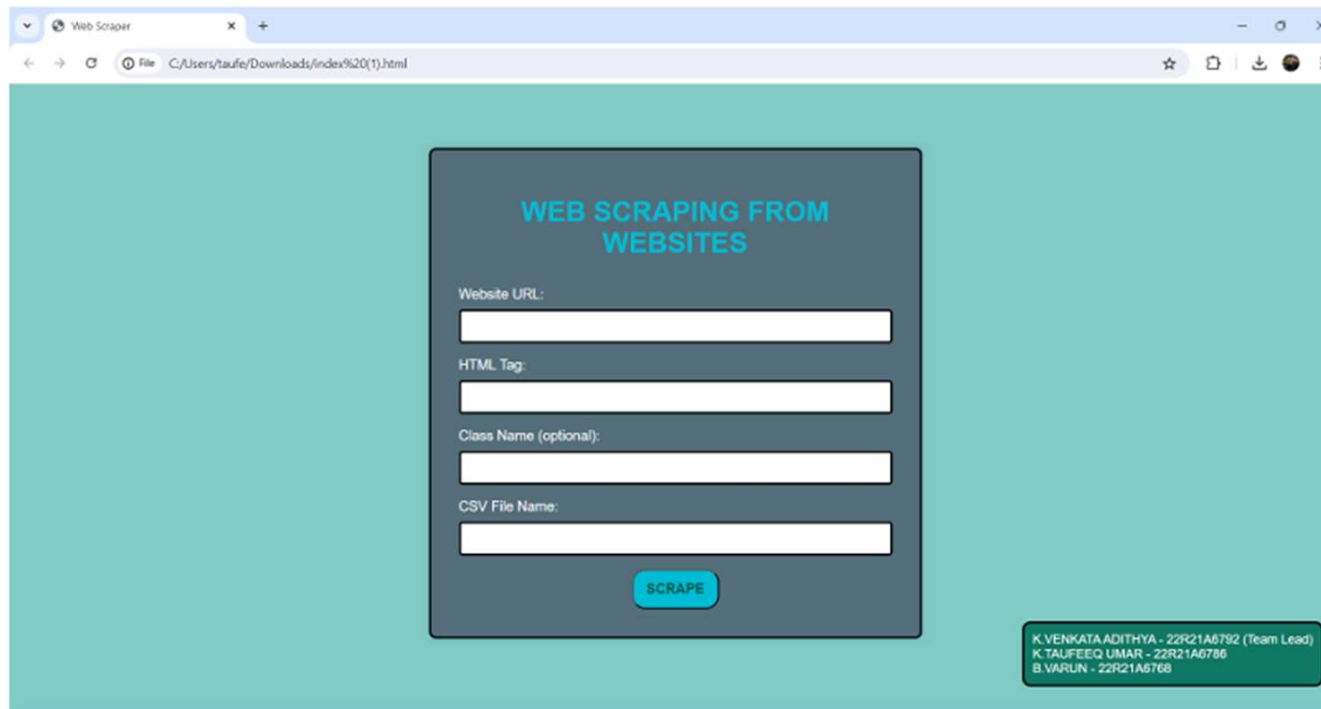


Figure 8.1: INTERFACE SCREENSHOT

Results

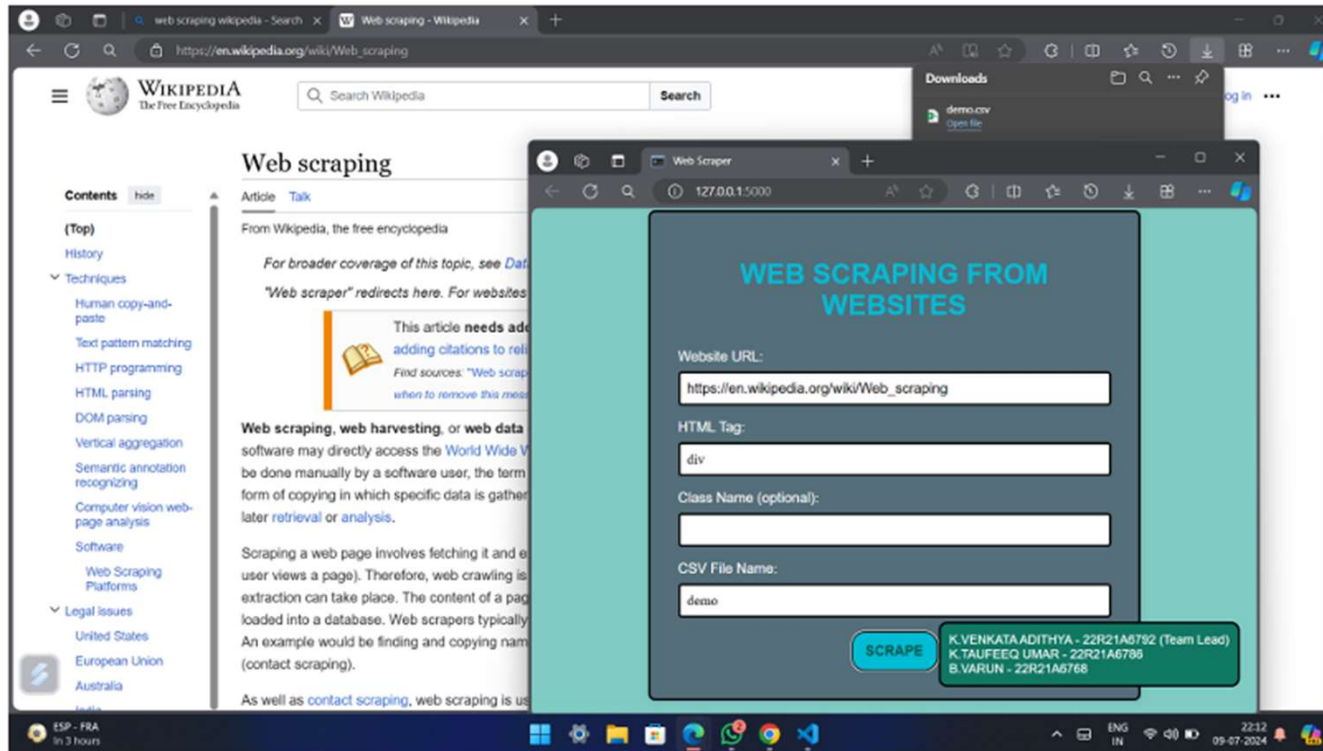


Figure 8.2: INTERFACE SCREENSHOT 2

Results

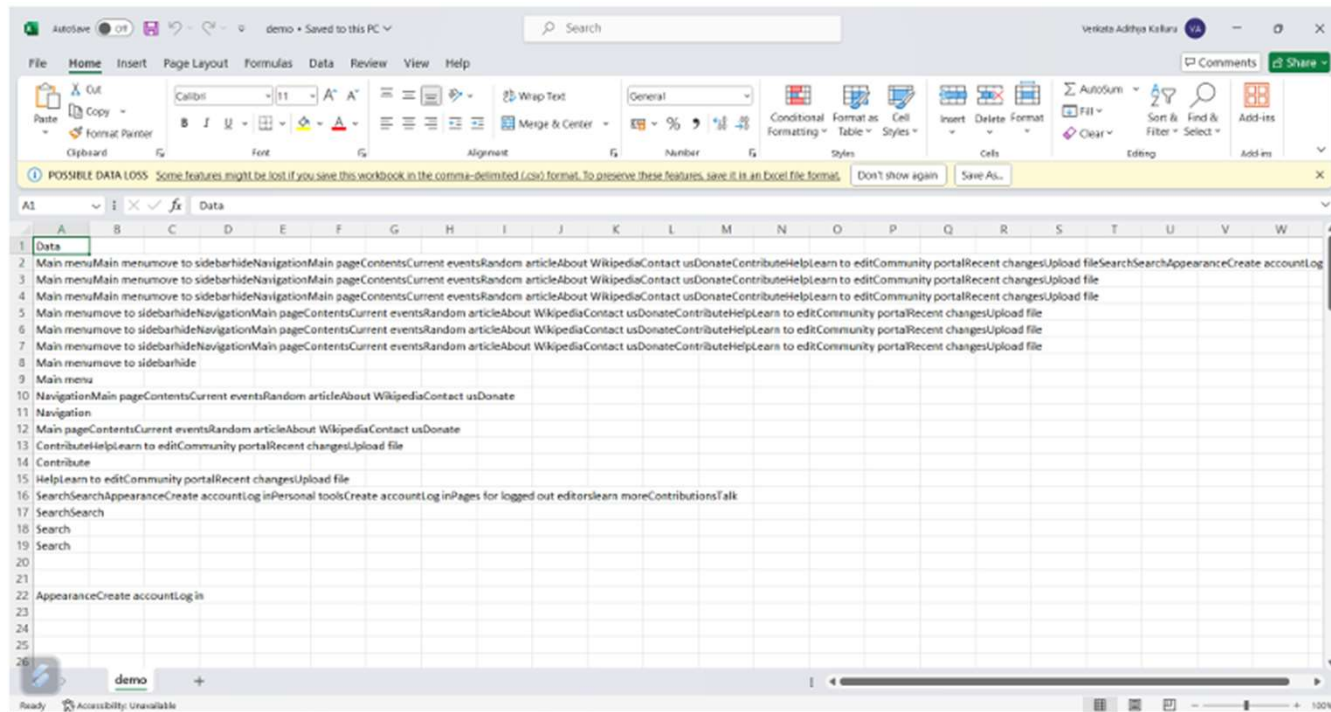


Figure 8.3: OUTPUT SCREENSHOT

Conclusion

web data extraction as a tool for information gathering, focusing on ethical practices and accessibility. By using the right methods and staying small-scale, we've shown how this technique can be effective for personal use and research without breaking the bank. By following legal and ethical guidelines, we've promoted responsible data collection and use, ensuring these valuable online resources stay available for everyone. web data extraction will undoubtedly play a more and more important role in various professional and academic fields.

References

Oxylabs - <https://oxylabs.io/blog/python-web-scraping>



Wikipedia - https://en.wikipedia.org/wiki/Web_scraping



WIKIPEDIA



THANK YOU

K.VENKATA ADITHYA

LINKEDIN : www.linkedin.com/in/venkataadithyakolluru

GITHUB : www.github.com/VenkataAdithyaKolluru

VENKATA ADITHYA KOLLURU