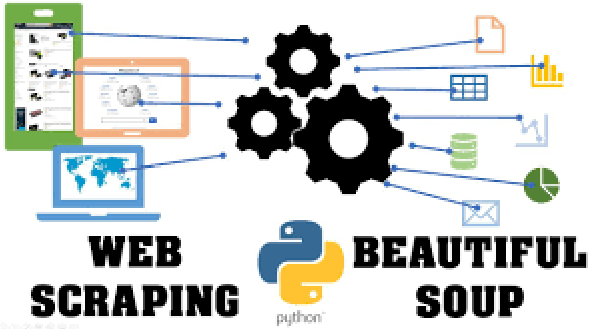
HTML Websites → Web Scraping → Data



Web Scraping: Introduction & Practical Using Python Flask

NIELIT Ropar

Web Scraping is the process of extracting and parsing data from websites in an automated fashion using a computer program

Web Scraping

HTML
HTML Websites
Web Scraping
Data

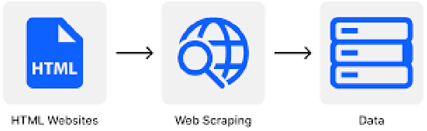
Introduction:

- Web scraping involves extracting data from websites for further processing and analysis.
- Often used for gathering data that isn’t readily available through APIs.

Key Tools:

- Python libraries such as BeautifulSoup and Scrapy.
- Automation tools like Selenium for dynamic content scraping.

What is Web Scraping?



- **Definition:**
  - The process of extracting data from websites.
- **Purpose:**
  - Automate the collection of information.
  - Convert unstructured data into a structured format.
- **Use Cases:**
  - Price comparison
  - Data analysis
  - Market research
  - News aggregation

Key Components of Web Scraping

HTML
HTML Websites
Web Scraping
Data

- **HTTP Requests:**
  - Sending GET/POST requests to web servers.
- **HTML Parsing:**
  - Extracting specific elements from the webpage.
- **Libraries:**
  - Python tools like **BeautifulSoup**, Scrapy, and Requests.
- **Output Formats:**
  - CSV, JSON, Databases, etc.

HTML Websites → Web Scraping → Data

Ethics and Legal Aspects

- **Guidelines:**
  - Check website’s Terms of Service.
  - Avoid scraping sensitive or copyrighted content.
  - Use polite scraping methods (e.g., rate limiting).
- **Tools to Avoid Detection:**
  - User-agent rotation.
  - Proxy servers.

Why Python for Web Scraping?

HTML
HTML Websites
Web Scraping
Data

- **Ease of Use:**
  - Simple syntax.
- **Libraries:**
  - Requests: For sending HTTP requests.
  - BeautifulSoup: For parsing HTML.
  - Selenium: For handling JavaScript-heavy websites.
- **Community Support:**
  - Extensive documentation and active forums.

Introduction to Flask

- **What is Flask?**
  - A lightweight web framework in Python.
- **Why Use Flask in Web Scraping?**
  - Build APIs to serve scraped data.
  - Create dashboards for data visualization.
  - Automate scraping tasks via web forms.

Workflow of Web Scraping

HTML
HTML Websites
Web Scraping
Data

1. **Identify the Target Website:**
    1. URL and specific data requirements.
2. **Send HTTP Requests:**
    1. Use requests.get() to fetch the webpage.
3. **Parse HTML Content:**
    1. Extract elements with BeautifulSoup.
4. **Store Data:**
    1. Save in a database or export as CSV/JSON.
5. **Build API with Flask:**
    1. Serve scraped data dynamically.

Hands-On Setup

HTML
HTML Websites
Web Scraping
Data

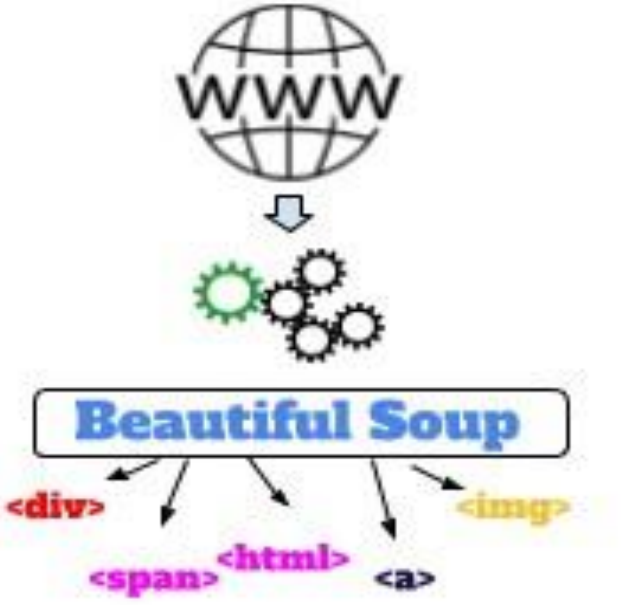
- **Install Required Libraries:**
  - pip install flask beautifulsoup4 requests
- **Basic Project Structure:**
  - app.py (Flask app)
  - scraper.py (Scraping logic)
  - templates/ (HTML files)

Code for Web Scraping Logic

HTML
HTML Websites
Web Scraping
Data

Example: Scraping Titles from a website

```
import requests
from bs4 import BeautifulSoup
def scrape_blog():
    url = https://example-blog.com
    response = requests.get(url)
    soup = BeautifulSoup(response.text, 'html.parser')
    titles = [title.text for title in soup.find_all('h2')]
    return titles
```



Web Scraping Code With Flask Integration

- Example:

```
from flask import Flask, jsonify
from scraper import scrape_blog
app = Flask(__name__)
@app.route('/’)
def home():
    data = scrape_blog()
    return jsonify(data)
if __name___ == '__main__’:
    app.run(debug=True)
```

Demo Time

HTML
HTML Websites
Web Scraping
Data

- **Steps:**
    - Run the Flask app: python app.py
    - Open the browser: http://127.0.0.1:5000/
    - View the scraped data in JSON format.

Visualization Ideas:

- Display the data in a table or graph.

Challenges in Web Scraping

- **Dynamic Content:**
  - JavaScript-heavy websites.
- **Anti-Scraping Mechanisms:**
  - CAPTCHAs, IP blocking.
- **Changing Website Structure:**
  - Frequent updates in HTML layout.

Best Practices

- Use proper headers (e.g., User-Agent).
- Respect robots.txt.
- Cache frequently scraped data.
- Handle errors and timeouts.

User-Agent

Most Common HTTP Headers for Web Scraping

Here’s a rundown of the most frequently used HTTP headers for web scraping:

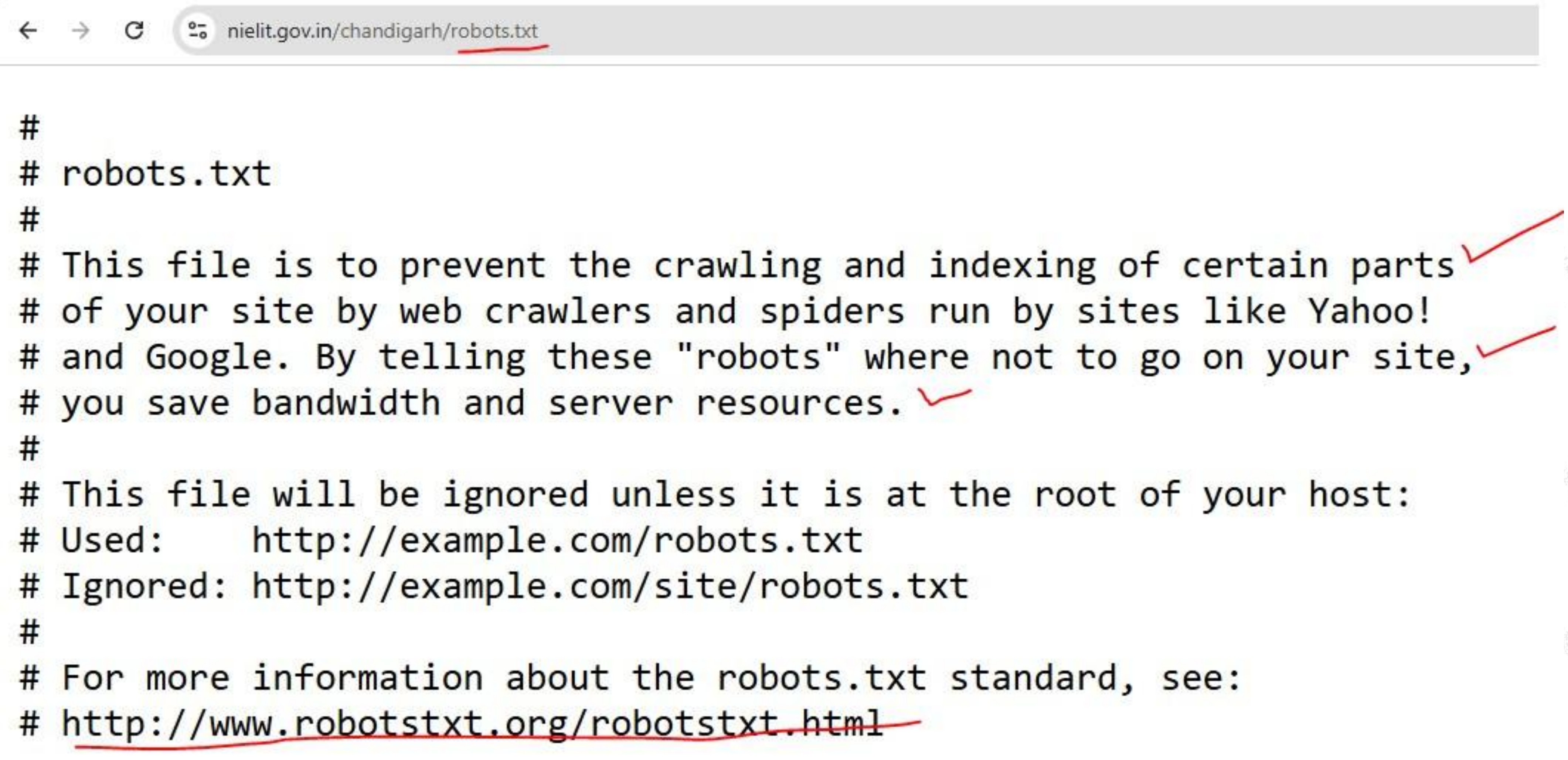
User-Agent

The User-Agent header identifies the browser or tool making the request. It’s one of the most important headers because most websites block non-browser user agents. Mimicking a real browser through this header can make your scraper look like legitimate traffic.

Example:

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36

Example of Robots.txt



nielit.gov.in/chandigarh/robots.txt

```
#
# robots.txt
#
# This file is to prevent the crawling and indexing of certain parts
# of your site by web crawlers and spiders run by sites like Yahoo!
# and Google. By telling these "robots" where not to go on your site,
# you save bandwidth and server resources.
#
# This file will be ignored unless it is at the root of your host:
# Used:    http://example.com/robots.txt
# Ignored: http://example.com/site/robots.txt
#
# For more information about the robots.txt standard, see:
# http://www.robotstxt.org/robotstxt.html
```

Why Use Cache For Web Scraping?

Caching increases performance and decreases computing costs by preventing redundancy. Here are a few reasons why you should use cache in web scraping:

- **Reduce Response Time**

Sending HTTP requests while scraping can be time-consuming, especially with rich and complex web pages. On the other hand, cached data are saved in memory storage and can be retrieved in fractions of a second. This boost in response time speeds up the web scraping process and **accelerates the development and debugging** cycles.

- **Reduce Server Load**

Instead of making repeated requests to the server for the same data, it can be retrieved from the cache. This reduces the number of requests sent, which prevents overloading the websites' servers and results in more ethical web scraping practices.

- **Reduce Consumed Bandwidth**

Using cached data can help minimize bandwidth usage by eliminating the number of repeated requests. This is particularly big when using residential proxies which charge by bandwidth and can be very expensive.

Use retry and timeout strategies

Another way to manage errors when web scraping with Python is to use retry and timeout strategies. Retry and timeout strategies are methods to handle network errors or delays, such as connection errors, server errors, or slow responses, when requesting a web page. Retry strategies allow you to repeat your web scraping requests a certain number of times or until a certain condition is met, in case of failures or errors. Timeout strategies allow you to specify a maximum time or limit for your web scraping requests, in case of delays or hangs. By using retry and timeout strategies, you can increase the success and efficiency of your web scraping requests, as well as avoid wasting resources or time.

Is Web Scraping Legal or Illegal?

- Web scraping legality depends on:

Illegal Scenarios:

- Violating terms of service.
- Bypassing security measures.
- Infringing copyright or privacy laws.
- Causing harm to website servers.

Legal Scenarios:

- Scraping publicly available data.
- Complying with terms of service.
- For research or personal use.
- Using open APIs.

Best Practices:

- Check terms of service.
- Seek permission if unsure.
- Respect robots.txt and server limits.
- Avoid sensitive data scraping.

Q&A

HTML Websites → Web Scraping → Data

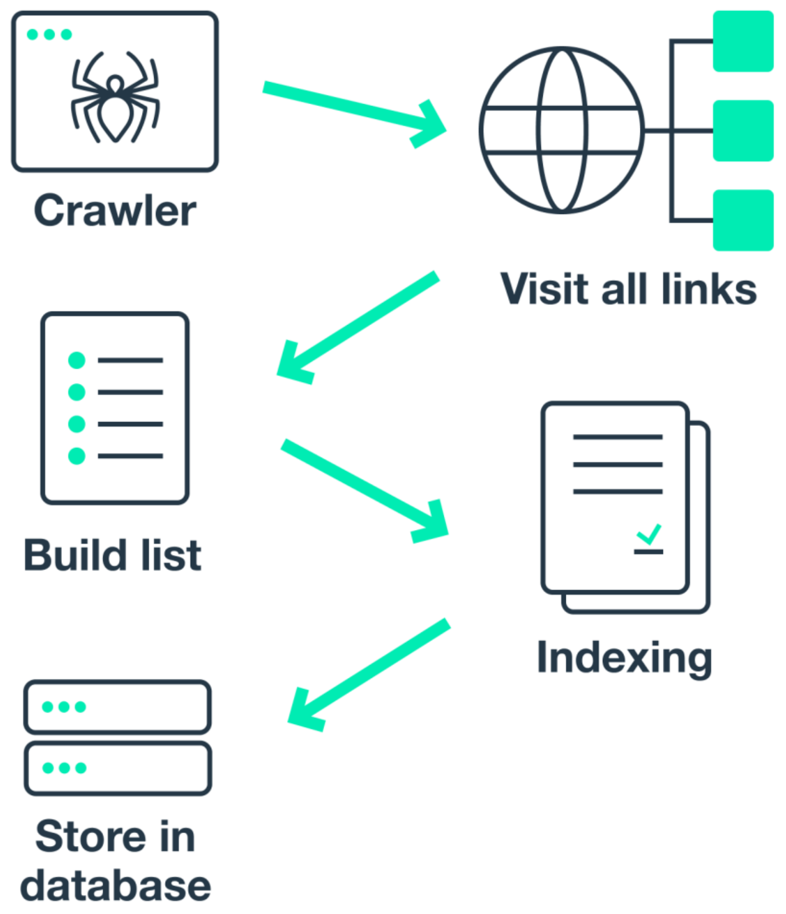
- **Let’s Discuss:**
- Any questions about setup or implementation?
- Real-world use cases for your projects.

Conclusion

HTML Websites → Web Scraping → Data

- **Key Takeaways:**
  - Python is a powerful tool for web scraping.
  - Flask makes it easy to build APIs and serve data.
  - Ethical scraping is critical for sustainable use.
- **Next Steps:**
  - Explore advanced techniques for web scraping like: Selenium, Scrapy.
  - Implement a complete data pipeline.

Web Crawler



Crawler

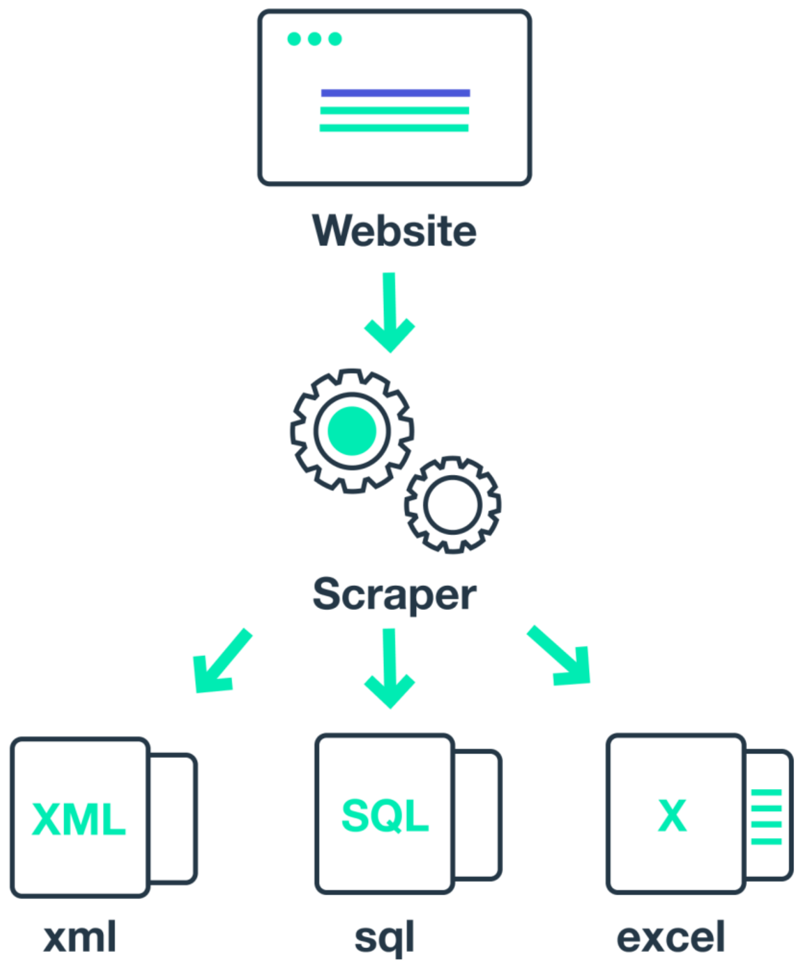
Visit all links

Build list

Indexing

Store in database

Web Scraping



Website

Scraper

XML

xml

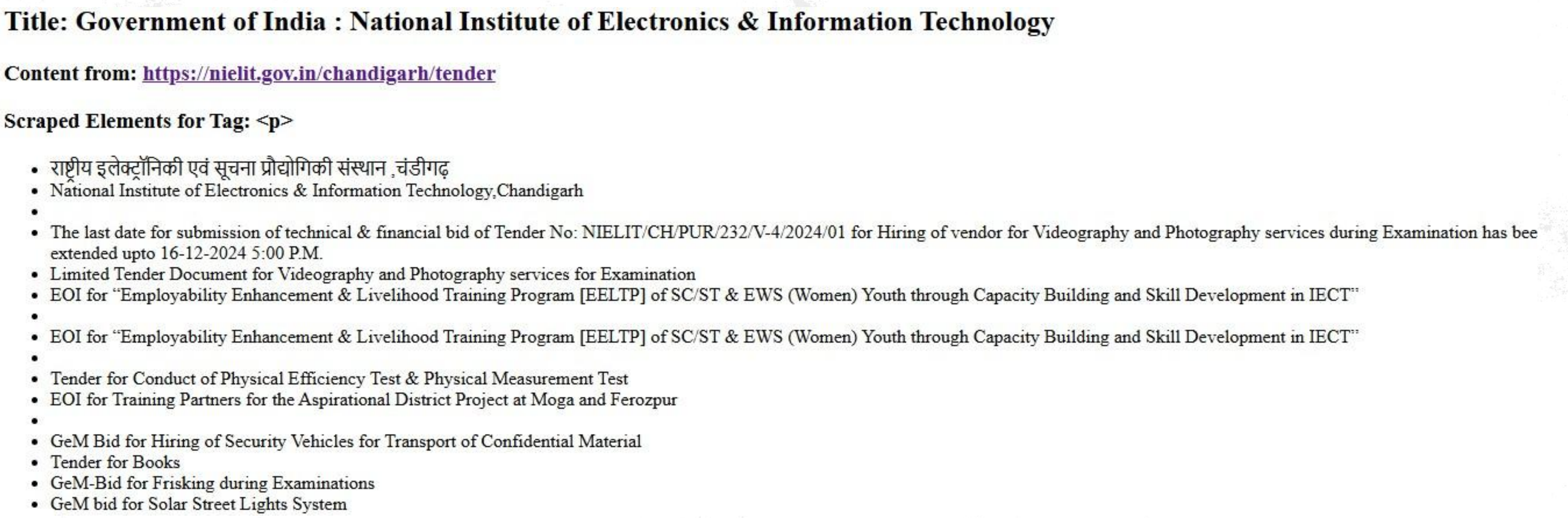
SQL

sql

X

excel

HTML
HTML Websites
Web Scraping
Data



Title: Government of India : National Institute of Electronics & Information Technology

Content from: https://nielit.gov.in/chandigarh/tender

Scraped Elements for Tag: <p>

- राष्ट्रीय इलेक्ट्रॉनिकी एवं सूचना प्रौद्योगिकी संस्थान ,चंडीगढ़
- National Institute of Electronics & Information Technology,Chandigarh
-
- The last date for submission of technical & financial bid of Tender No: NIELIT/CH/PUR/232/V-4/2024/01 for Hiring of vendor for Videography and Photography services during Examination has bee extended upto 16-12-2024 5:00 P.M.
- Limited Tender Document for Videography and Photography services for Examination
- EOI for "Employability Enhancement & Livelihood Training Program [EELTP] of SC/ST & EWS (Women) Youth through Capacity Building and Skill Development in IECT"
-
- EOI for "Employability Enhancement & Livelihood Training Program [EELTP] of SC/ST & EWS (Women) Youth through Capacity Building and Skill Development in IECT"
-
- Tender for Conduct of Physical Efficiency Test & Physical Measurement Test
- EOI for Training Partners for the Aspirational District Project at Moga and Ferozpur
-
- GeM Bid for Hiring of Security Vehicles for Transport of Confidential Material
- Tender for Books
- GeM-Bid for Frisking during Examinations
- GeM bid for Solar Street Lights System

LIVE DEMO:

https://web-scraping-7mmm.onrender.com/

Flask

python™

HTML Websites → Web Scraping → Data

BeautifulSoup

Hugging Face

Create Live Project

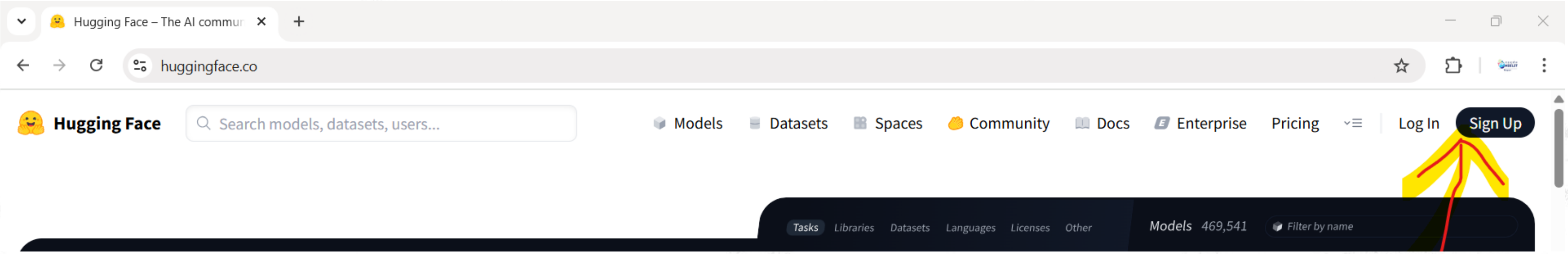
NIELIT Ropar

रा.इ.सू.प्रौ.सं
NIELIT
Deemed to be University

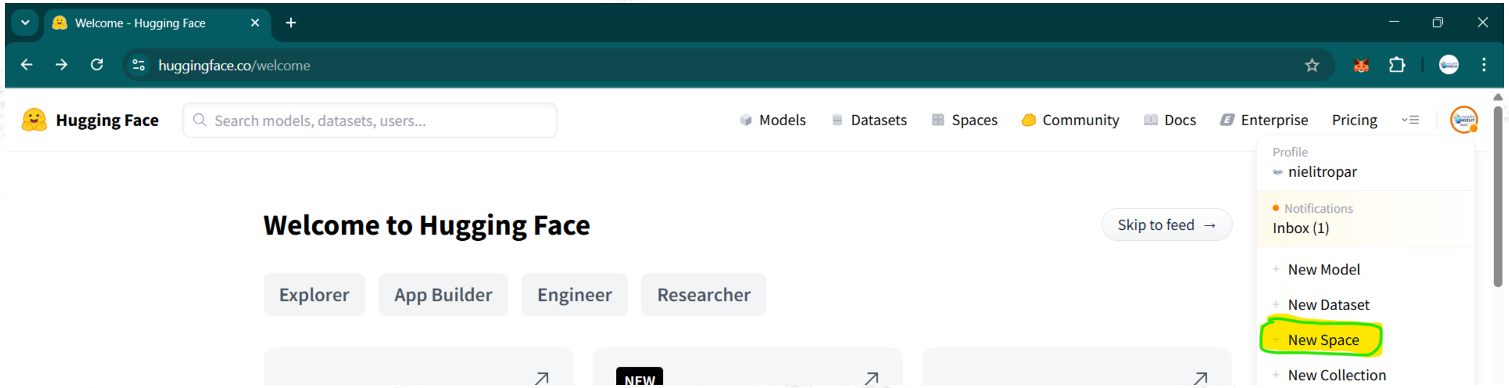
Web Scraping is the process of extracting and parsing data from websites in an automated fashion using a computer program

Create Webscraping website using Python BeautifulSoup, Flask Framework On https://huggingface.co/

- **Set Up HuggingFace Project (space):**
  - Sign up or Login on https://huggingface.co/

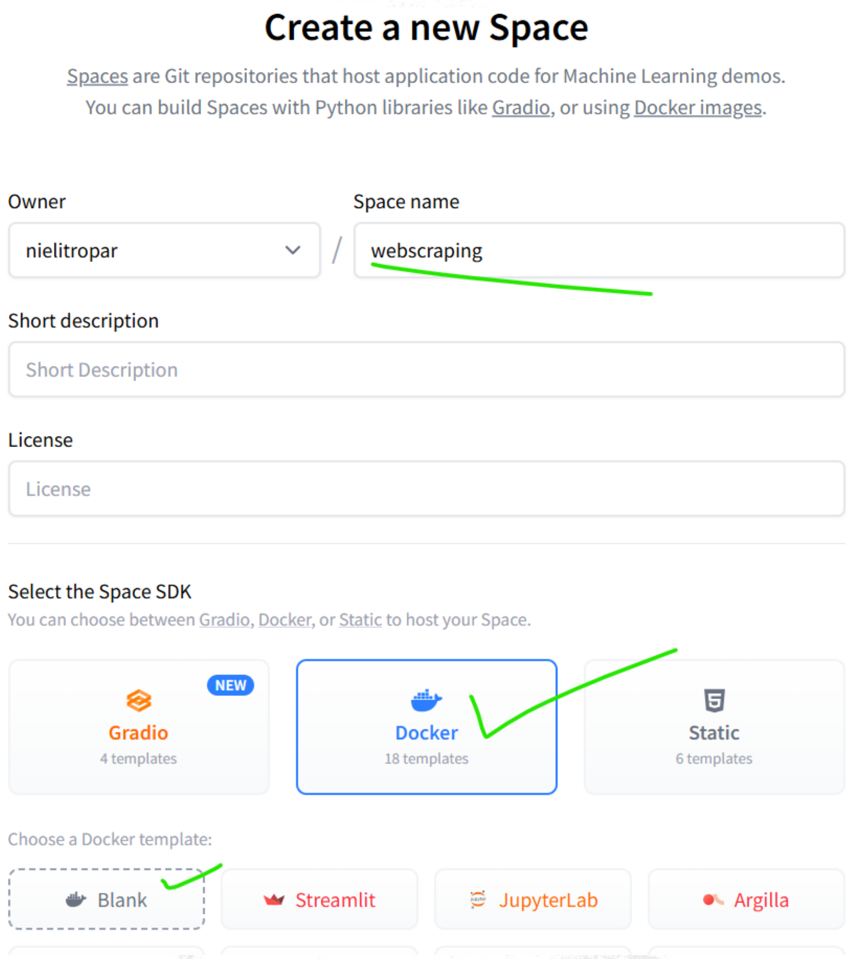


  - Go to https://huggingface.co/ homepage after logging in and click on **New Space**.



Create Webscraping website using Python BeautifulSoup, Flask Framework On https://huggingface.co/

- Name your Space and set SDK Docker keep template blank and create Space



Create a new Space

Spaces are Git repositories that host application code for Machine Learning demos.
You can build Spaces with Python libraries like Gradio, or using Docker images.

Owner: nielitropar / Space name: webscraping

Short description: Short Description

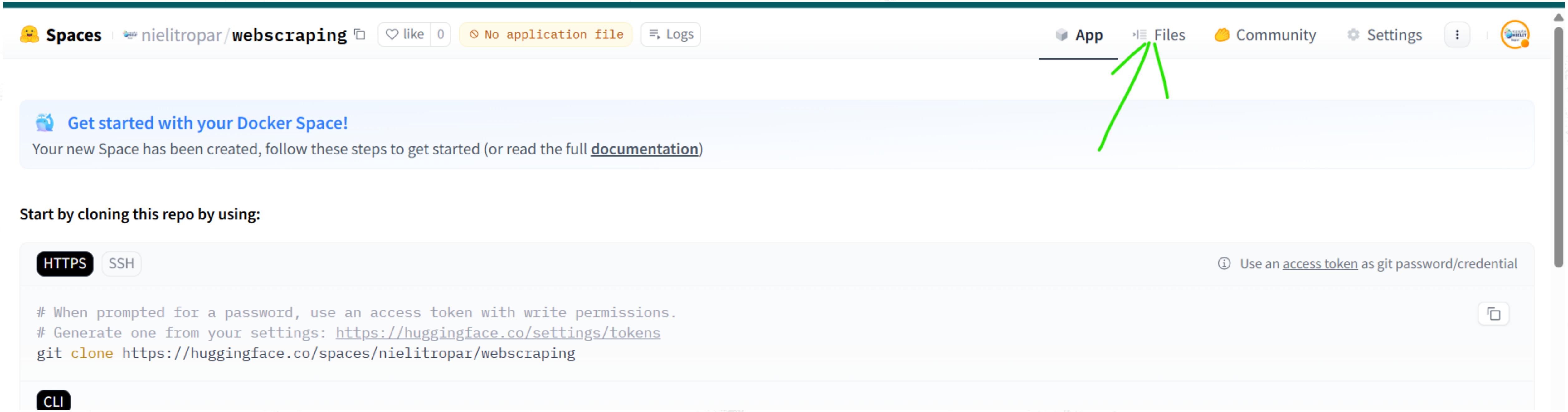
License: License

Select the Space SDK
You can choose between Gradio, Docker, or Static to host your Space.

Choose a Docker template:

Create Webscraping website using Python BeautifulSoup, Flask Framework On https://huggingface.co/

➢ Click on Files tab



Get started with your Docker Space!

Your new Space has been created, follow these steps to get started (or read the full documentation)

Start by cloning this repo by using:

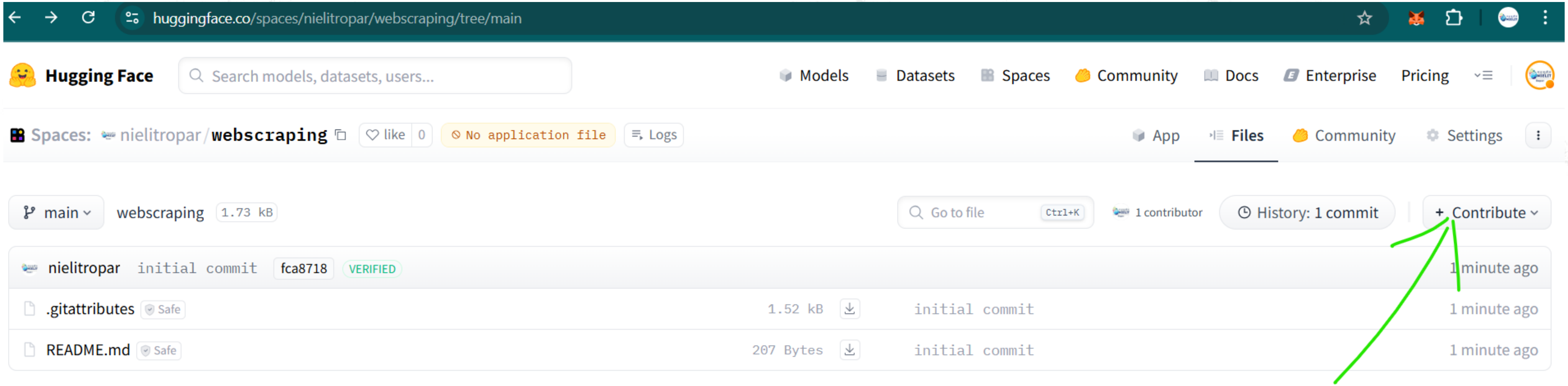
Use an access token as git password/credential

```
# When prompted for a password, use an access token with write permissions.
# Generate one from your settings: https://huggingface.co/settings/tokens
git clone https://huggingface.co/spaces/nielitropar/webscraping
```

Create Webscraping website using Python BeautifulSoup, Flask Framework On https://huggingface.co/

HTML Websites → Web Scraping → Data

1. Click on Contribute and Create a new file
2. Name your file
3. Add file code
4. Press Commit new file to main


huggingface.co/spaces/nielitropar/webscraping/tree/main

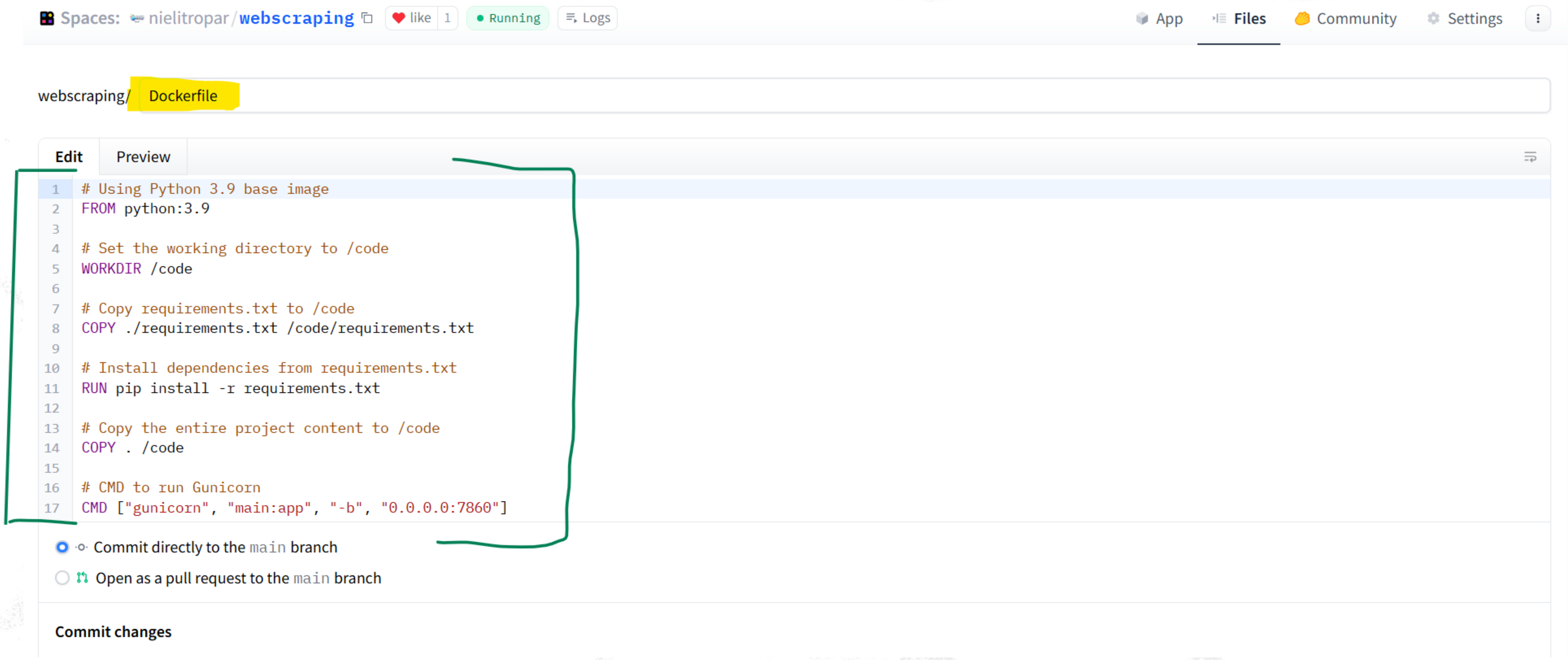
webscraping 1.73 kB

nielitropar initial commit fca8718 VERIFIED 1 minute ago

.gitattributes	Safe	1.52 kB	initial commit	1 minute ago
README.md	Safe	207 Bytes	initial commit	1 minute ago

Docker File

Create a new file name is Dockerfile and add following commands in it and press commit



webscraping/ Dockerfile

```
# Using Python 3.9 base image
FROM python:3.9

# Set the working directory to /code
WORKDIR /code

# Copy requirements.txt to /code
COPY ./requirements.txt /code/requirements.txt

# Install dependencies from requirements.txt
RUN pip install -r requirements.txt

# Copy the entire project content to /code
COPY . /code

# CMD to run Gunicorn
CMD ["gunicorn", "main:app", "-b", "0.0.0.0:7860"]
```

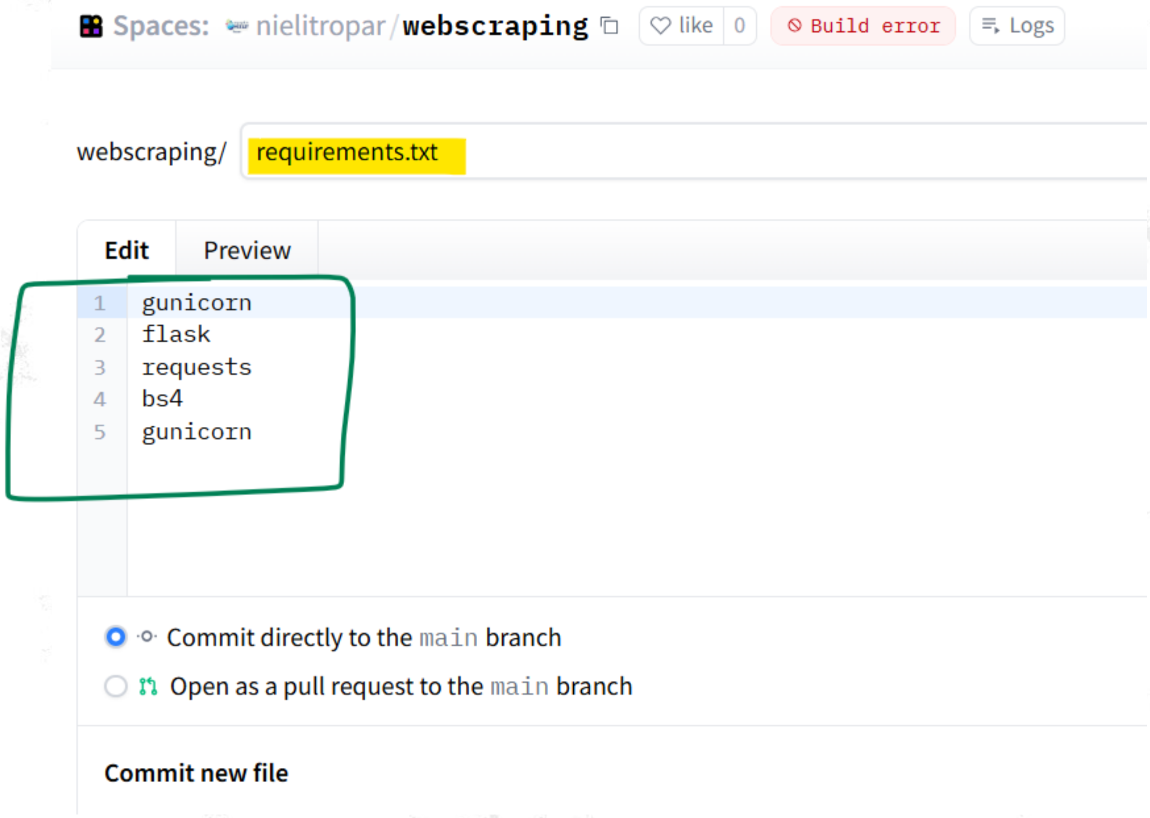
● Commit directly to the main branch

○ Open as a pull request to the main branch

Commit changes

requirements.txt

- Create a new file name it requirements.txt
- Add list of all required libraries in requirements.txt file.
- Press commit



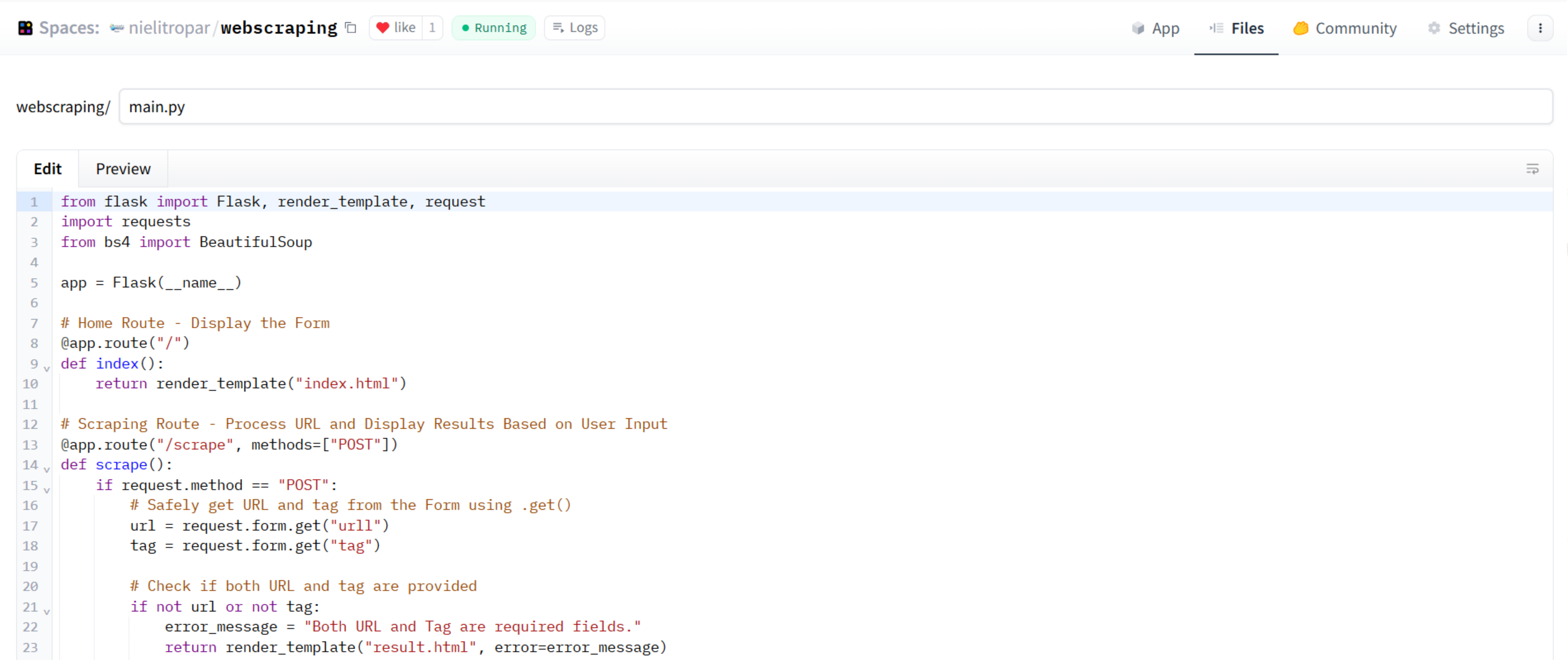
webscraping/ requirements.txt

```
gunicorn
flask
requests
bs4
gunicorn
```

● Commit directly to the main branch

○ Open as a pull request to the main branch

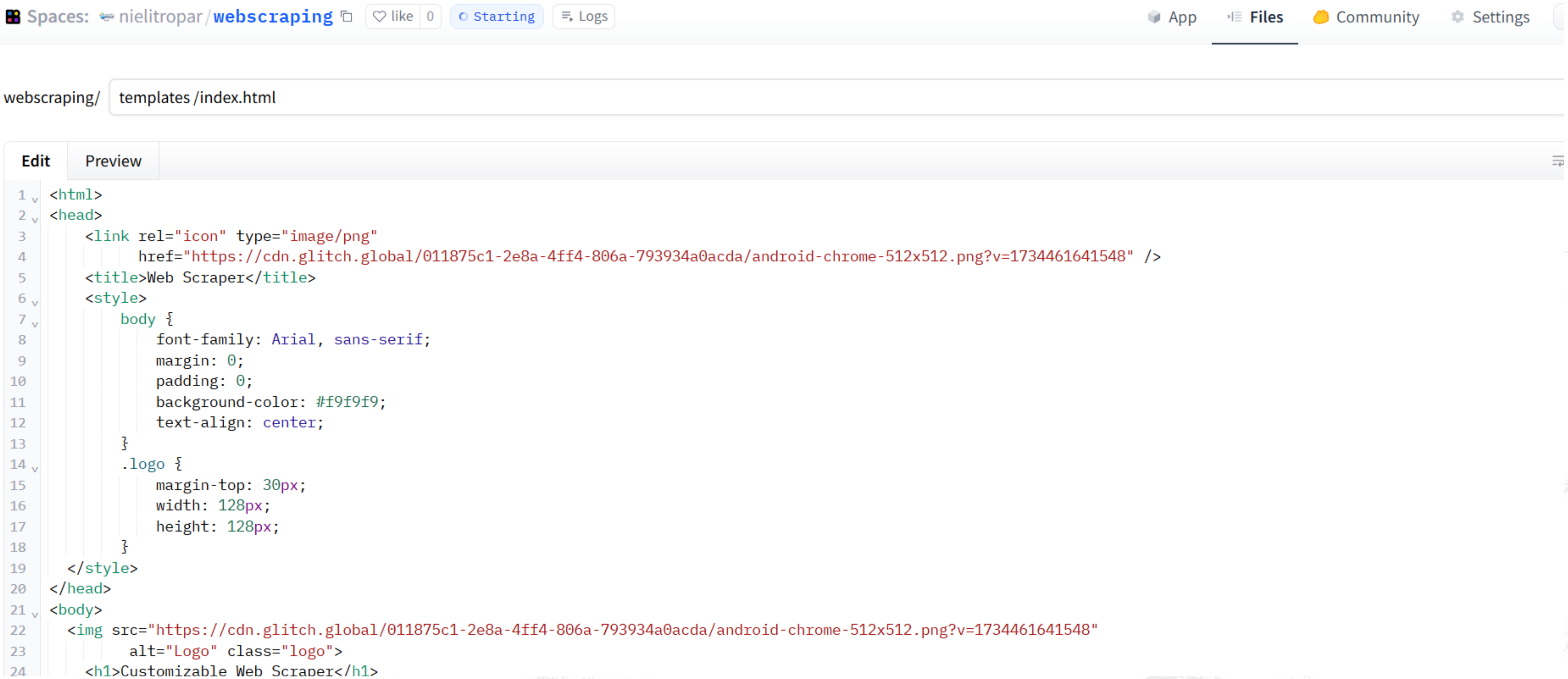
main.py



webscraping/ main.py

```
from flask import Flask, render_template, request
import requests
from bs4 import BeautifulSoup

app = Flask(__name__)

# Home Route - Display the Form
@app.route("/")
def index():
    return render_template("index.html")

# Scraping Route - Process URL and Display Results Based on User Input
@app.route("/scrape", methods=["POST"])
def scrape():
    if request.method == "POST":
        # Safely get URL and tag from the Form using .get()
        url = request.form.get("urll")
        tag = request.form.get("tag")

        # Check if both URL and tag are provided
        if not url or not tag:
            error_message = "Both URL and Tag are required fields."
            return render_template("result.html", error=error_message)
```

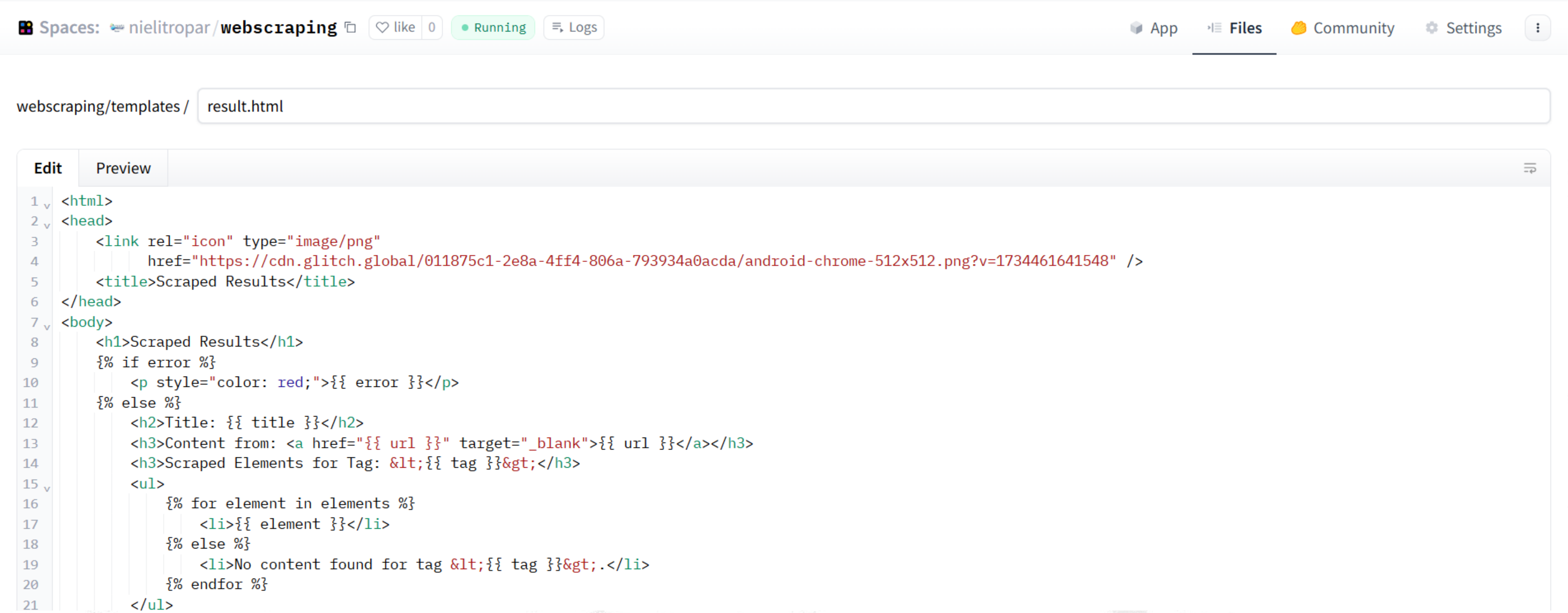
templates/index.html



webscraping/ templates /index.html

```html
<html>
<head>
    <link rel="icon" type="image/png"
          href="https://cdn.glitch.global/011875c1-2e8a-4ff4-806a-793934a0acda/android-chrome-512x512.png?v=1734461641548" />
    <title>Web Scraper</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
            background-color: #f9f9f9;
            text-align: center;
        }
        .logo {
            margin-top: 30px;
            width: 128px;
            height: 128px;
        }
  </style>
</head>
<body>
  <img src="https://cdn.glitch.global/011875c1-2e8a-4ff4-806a-793934a0acda/android-chrome-512x512.png?v=1734461641548"
         alt="Logo" class="logo">
    <h1>Customizable Web Scraper</h1>
```

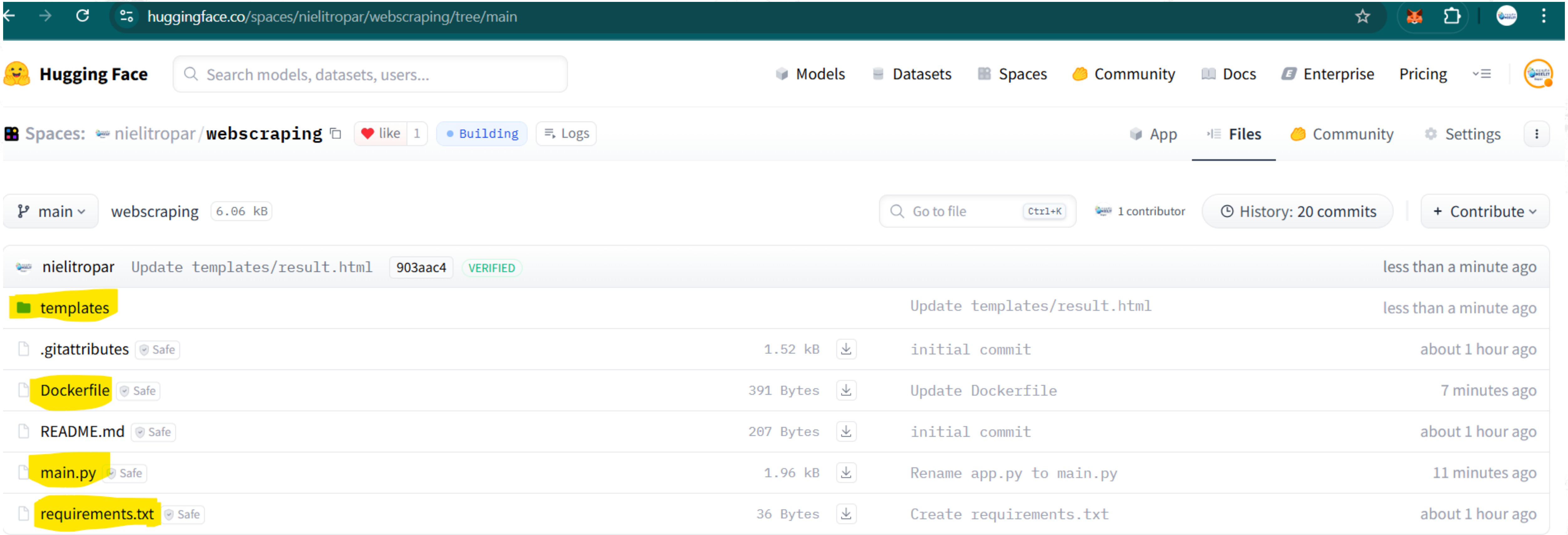
templates/result.html



webscraping/templates / result.html

```html
<html>
<head>
    <link rel="icon" type="image/png"
          href="https://cdn.glitch.global/011875c1-2e8a-4ff4-806a-793934a0acda/android-chrome-512x512.png?v=1734461641548" />
    <title>Scraped Results</title>
</head>
<body>
    <h1>Scraped Results</h1>
    {% if error %}
        <p style="color: red;">{{ error }}</p>
    {% else %}
        <h2>Title: {{ title }}</h2>
        <h3>Content from: <a href="{{ url }}" target="_blank">{{ url }}</a></h3>
        <h3>Scraped Elements for Tag: &lt;{{ tag }}&gt;</h3>
        <ul>
            {% for element in elements %}
                <li>{{ element }}</li>
            {% else %}
                <li>No content found for tag &lt;{{ tag }}&gt;.</li>
            {% endfor %}
        </ul>
```

File/Folder Structure

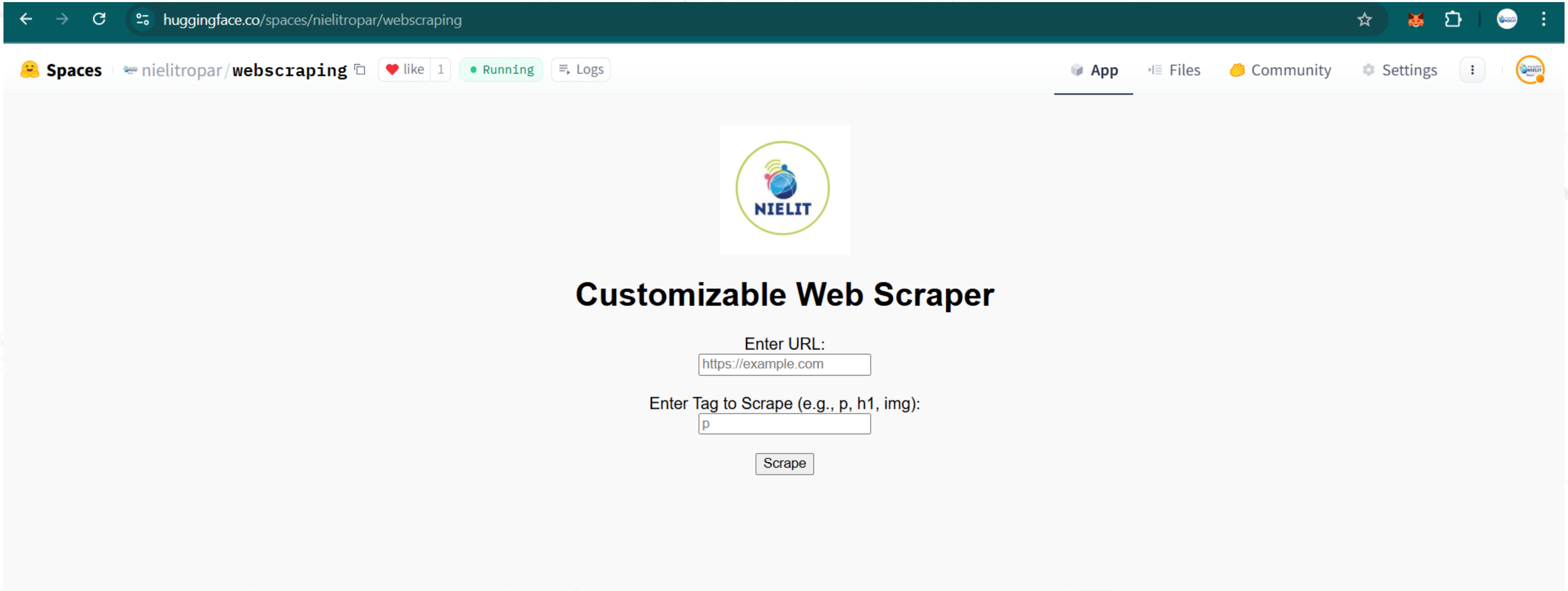


webscraping 6.06 kB

nielitropar Update templates/result.html 903aac4 VERIFIED less than a minute ago

Name	Size	Last commit	Updated
templates		Update templates/result.html	less than a minute ago
.gitattributes Safe	1.52 kB	initial commit	about 1 hour ago
Dockerfile Safe	391 Bytes	Update Dockerfile	7 minutes ago
README.md Safe	207 Bytes	initial commit	about 1 hour ago
main.py Safe	1.96 kB	Rename app.py to main.py	11 minutes ago
requirements.txt Safe	36 Bytes	Create requirements.txt	about 1 hour ago

Click on App to run Webapp



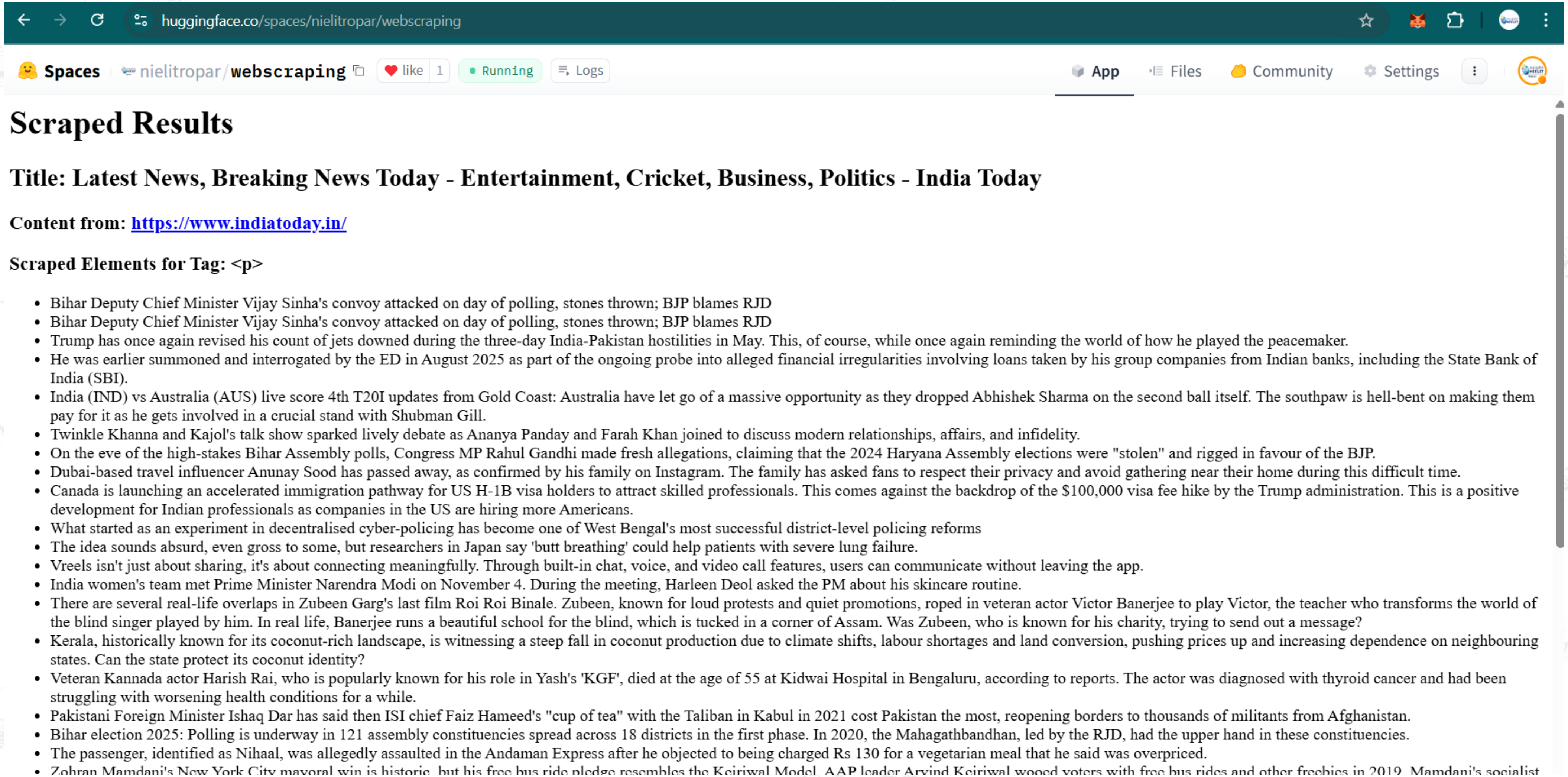
Customizable Web Scraper

Enter URL: https://example.com

Enter Tag to Scrape (e.g., p, h1, img): p

Scrape

Enjoy your own Web Scraping App



Scraped Results

Title: Latest News, Breaking News Today - Entertainment, Cricket, Business, Politics - India Today

Content from: https://www.indiatoday.in/

Scraped Elements for Tag: <p>

- Bihar Deputy Chief Minister Vijay Sinha's convoy attacked on day of polling, stones thrown; BJP blames RJD
- Bihar Deputy Chief Minister Vijay Sinha's convoy attacked on day of polling, stones thrown; BJP blames RJD
- Trump has once again revised his count of jets downed during the three-day India-Pakistan hostilities in May. This, of course, while once again reminding the world of how he played the peacemaker.
- He was earlier summoned and interrogated by the ED in August 2025 as part of the ongoing probe into alleged financial irregularities involving loans taken by his group companies from Indian banks, including the State Bank of India (SBI).
- India (IND) vs Australia (AUS) live score 4th T20I updates from Gold Coast: Australia have let go of a massive opportunity as they dropped Abhishek Sharma on the second ball itself. The southpaw is hell-bent on making them pay for it as he gets involved in a crucial stand with Shubman Gill.
- Twinkle Khanna and Kajol's talk show sparked lively debate as Ananya Panday and Farah Khan joined to discuss modern relationships, affairs, and infidelity.
- On the eve of the high-stakes Bihar Assembly polls, Congress MP Rahul Gandhi made fresh allegations, claiming that the 2024 Haryana Assembly elections were "stolen" and rigged in favour of the BJP.
- Dubai-based travel influencer Anunay Sood has passed away, as confirmed by his family on Instagram. The family has asked fans to respect their privacy and avoid gathering near their home during this difficult time.
- Canada is launching an accelerated immigration pathway for US H-1B visa holders to attract skilled professionals. This comes against the backdrop of the $100,000 visa fee hike by the Trump administration. This is a positive development for Indian professionals as companies in the US are hiring more Americans.
- What started as an experiment in decentralised cyber-policing has become one of West Bengal's most successful district-level policing reforms
- The idea sounds absurd, even gross to some, but researchers in Japan say 'butt breathing' could help patients with severe lung failure.
- Vreels isn't just about sharing, it's about connecting meaningfully. Through built-in chat, voice, and video call features, users can communicate without leaving the app.
- India women's team met Prime Minister Narendra Modi on November 4. During the meeting, Harleen Deol asked the PM about his skincare routine.
- There are several real-life overlaps in Zubeen Garg's last film Roi Roi Binale. Zubeen, known for loud protests and quiet promotions, roped in veteran actor Victor Banerjee to play Victor, the teacher who transforms the world of the blind singer played by him. In real life, Banerjee runs a beautiful school for the blind, which is tucked in a corner of Assam. Was Zubeen, who is known for his charity, trying to send out a message?
- Kerala, historically known for its coconut-rich landscape, is witnessing a steep fall in coconut production due to climate shifts, labour shortages and land conversion, pushing prices up and increasing dependence on neighbouring states. Can the state protect its coconut identity?
- Veteran Kannada actor Harish Rai, who is popularly known for his role in Yash's 'KGF', died at the age of 55 at Kidwai Hospital in Bengaluru, according to reports. The actor was diagnosed with thyroid cancer and had been struggling with worsening health conditions for a while.
- Pakistani Foreign Minister Ishaq Dar has said then ISI chief Faiz Hameed's "cup of tea" with the Taliban in Kabul in 2021 cost Pakistan the most, reopening borders to thousands of militants from Afghanistan.
- Bihar election 2025: Polling is underway in 121 assembly constituencies spread across 18 districts in the first phase. In 2020, the Mahagathbandhan, led by the RJD, had the upper hand in these constituencies.
- The passenger, identified as Nihaal, was allegedly assaulted in the Andaman Express after he objected to being charged Rs 130 for a vegetarian meal that he said was overpriced.
- Zohran Mamdani's New York City mayoral win is historic, but his free bus ride pledge resembles the Kejriwal Model. AAP leader Arvind Kejriwal wooed voters with free bus rides and other freebies in 2019. Mamdani's socialist

Live Demo

HTML Websites → Web Scraping → Data

https://nielitropar-webscraping.hf.space/

Source Code

- https://huggingface.co/spaces/nielitropar/webscraping/tree/main
- https://huggingface.co/spaces/nielitropar/webscraping
- https://nielitropar-webscraping.hf.space/