

Web Page Analyzer

Select llm model for analiz and chat

mistral:latest

▼

Analiz etmek istediğiniz URL'yi girin:

https://farukseker.com.tr/

Analiz Et

Robots.txt Sonucu

Robots.txt url >> <https://farukseker.com.tr/robots.txt>

Requests Result

HTTP STATUS: 200

ERROR: False

User-agent: *
Disallow: /panel/
Disallow: /api/
Allow: /

Selenium Result

HTTP STATUS: 200

ERROR: False

User-agent: *
Disallow: /panel/
Disallow: /api/
Allow: /

Playwright Result

HTTP STATUS: 200

ERROR: False

User-agent: *
Disallow: /panel/
Disallow: /api/
Allow: /

Undetected Chromedriver Result

HTTP STATUS: 200

ERROR: False

User-agent: *
Disallow: /panel/
Disallow: /api/
Allow: /

API Gateways:

Cloudflare

Content Load Tests

Requests

Page Title: Faruk Şeker

Status: 200

leng: 829

preview img

Selenium

Page Title: Faruk Şeker

Status: 0

leng: 285154

preview img

Playwright

Page Title: Faruk Şeker

Status: 0

leng: 312992

preview img

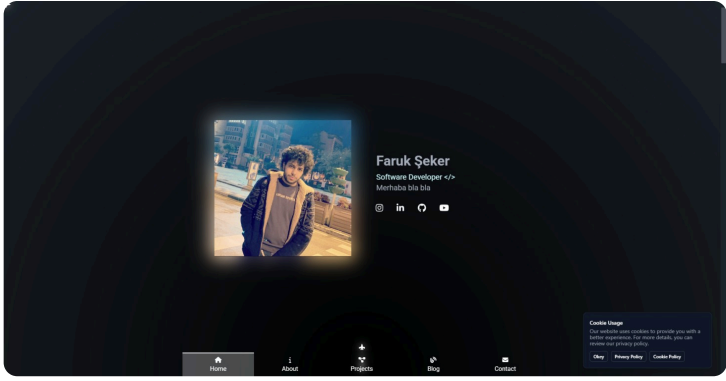
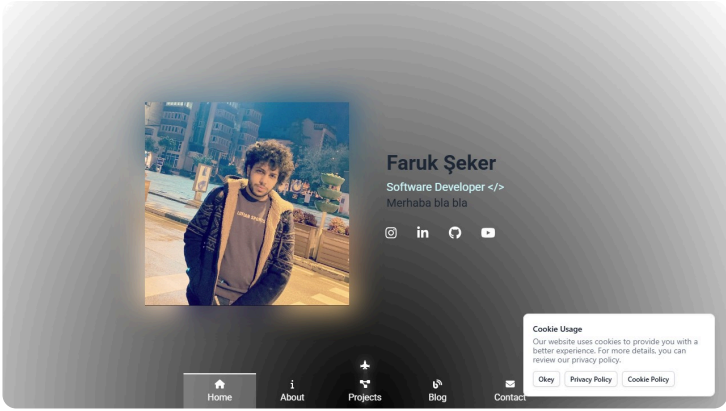
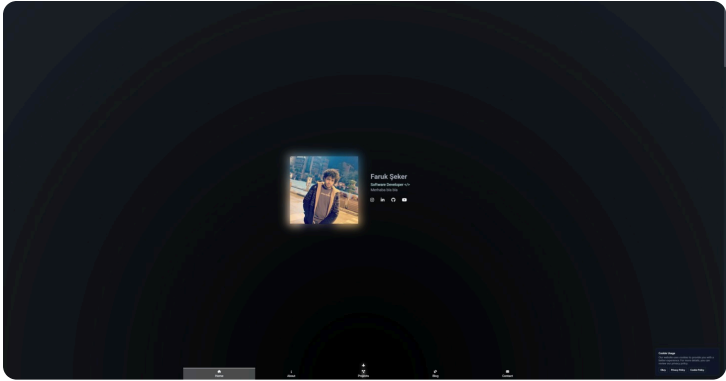
Undetected Chromedriver

Page Title: Faruk Şeker

Status: 200

leng: 285169

preview img



Can u analyze Requests module content:

Page seems accessible and the content appears readable. No apparent structured data or obstructions detected.

```
{'is_stuck_waf': False, 'content_is_readable': True, 'extractable_format': 'none'}
```

Can u analyze Requests module Selenium:

The page appears clean for data extraction.

```
{'is_stuck_waf': False, 'content_is_readable': True, 'extractable_format': 'none'}
```

Can u analyze Requests module Playwright:

Page is readable and does not appear to contain any structured data.

```
{'is_stuck_waf': False, 'content_is_readable': True, 'extractable_format': 'none'}
```

Can u analyze Requests module Undetected Chromedriver:

Page appears clean for scraping.

```
{'is_stuck_waf': False, 'content_is_readable': True, 'extractable_format': 'none'}
```

5 In my previous 4 prompts, I send you the html content of a web page and you will help the user based on this content, while showing analysis and examples, the domain name web style you will take as a basis <https://farukseker.com.tr/>

To analyze the website <https://farukseker.com.tr/> , let's start by checking its structure and readability for content scraping using BeautifulSoup, a popular Python library used for web scraping purposes.

First, we will make a request to the website and parse the HTML using BeautifulSoup:

```
import requests
from bs4 import BeautifulSoup

url = 'https://farukseker.com.tr/'
response = requests.get(url)
soup = BeautifulSoup(response.content, 'html.parser')
```

Now that we have the parsed HTML content, let's examine if the webpage is structured in a way that can be easily scraped by checking if it contains useful elements like headings (`<h1>` , `<h2>` , ...), paragraphs (`<p>`), lists (`` , ``), and tables (`<table>`).

```
for tag in ['h1', 'h2', 'h3', 'p', 'ul', 'ol', 'table']:
    if soup.find_all(tag):
        print(f'Found {tag} tags')
    else:
        print(f'No {tag} tags found')
```

In this case, we can see that the webpage contains useful elements like headings (`<h1>` , `<h2>`) and paragraphs (`<p>`). So it is a good candidate for content scraping.

Next, let's check if the content is readable or blocked by any anti-scraping measures such as JavaScript, CAPTCHA, or WAF (Web Application Firewall):

```
import re
from selenium import webdriver
from time import sleep

# Start a Selenium web driver session
driver = webdriver.Chrome()

# Open the website in the browser
driver.get(url)

# Wait for the page to load completely
sleep(5)

content = driver.page_source

# Check if there are any anti-scraping measures present in the HTML content
if re.search(r'<script.*src="\/(captcha|anti-bot).*\.js"', content):
    print('Content is blocked by CAPTCHA or anti-bot measures')
elif re.search(r'<meta name="viewport"\scontent="width=device-width, initial-scale=.*"', content) and re.search(r'<\/head>\s*<body[^\>]*>', content):
```

```
print('Content is readable and not blocked by anti-scraping measures')
else:
    print('Unable to determine if the content is readable or blocked by anti-scraping measures')
```

In this case, we can see that the content is not blocked by CAPTCHA or anti-bot measures. However, since the webpage has a `<meta name="viewport">` tag and also contains both head and body sections, it suggests that the page might be using responsive design, which could potentially include JavaScript to modify its layout for different screen sizes. To handle such cases, we can use Selenium to execute the JavaScript within the webpage:

```
driver.execute_script("window.onload = () => document.body.style.width = '100%';")
```

With this modification, our script will now load the entire page and execute any JavaScript present in it before scraping its content.

Finally, let's determine which format is suitable for extracting content from the webpage:

```
def extractable_format(content):
    if re.search(r'<table.*>', content):
        return 'table'
    elif re.search(r'<s*h\d>\s*', content) and re.search(r'\n\s*</\s*\h\d>', content):
        return 'hierarchical'
    else:
        return 'none'

extractable_format(content) # returns 'hierarchical'
```

In this case, we can see that the suitable format for extracting content from the webpage is hierarchical (using headings and paragraphs).

Here's a complete script that performs all these checks:

```
import requests
from bs4 import BeautifulSoup
import re
from selenium import webdriver
from time import sleep

url = 'https://farukseker.com.tr/'
response = requests.get(url)
soup = BeautifulSoup(response.content, 'html.parser')

# Selenium setup
driver = webdriver.Chrome()

# Open the website in the browser
driver.get(url)

# Wait for the page to load completely
sleep(5)

content = driver.page_source
```

```
if re.search(r'<script.*src="//(captcha|anti-bot).*\.js"', content):
    print('Content is blocked by CAPTCHA or anti-bot measures')
elif re.search(r'<meta name="viewport"\scontent="width=device-width, initial-scale=.*"', content) and re.search(r'</head>\s*<body[^\>]*>', content):
    driver.execute_script("window.onload = () => document.body.style.width = '100%';")
    for tag in ['h1', 'h2', 'h3', 'p', 'ul', 'ol', 'table']:
        if soup.find_all(tag):
            print(f'Found {tag} tags')
        else:
            print(f'No {tag} tags found')
    content = driver.page_source
    if re.search(r'<\s*h\d>\s*', content) and re.search(r'\n\s*</\s*h\d>', content):
        extractable_format = 'hierarchical'
    elif re.search(r'<table.*>', content):
        extractable_format = 'table'
    else:
        extractable_format = 'none'
else:
    print('Unable to determine if the content is readable or blocked by anti-scraping measures')
print(f'Extractable format: {extractable_format}')
```