RESOLUSI : Rekayasa Teknik Informatika dan Informasi
ISSN 2745-7966 (Media Online)
Vol 3, No 2, November 2022
Hal 63-68
https://djournals.com/resolusi

Impementasi Web Scraping Pada OJS Dengan Metode CSS Selector

Agus Purnomo

Fakultas Tarbiyah dan Ilmu Keguruan, Pendidikan Agama Islam, Institute Agama Islam Negeri Salatiga, Salatiga, Indonesia
Email: agus.purnomo@iainsalatiga.ac.id
Email Penulis Korespondensi: agus.purnomo@iainsalatiga.ac.id

Abstrak−Sumber rujukan dalam penelitian dan penulisan artikel ilmiah adalah jurnal. Untuk jurnal internasional bereputasi sudah banyak perangkat lunak yang dikembangkan untuk menarik meta data guna pemetaan penelitian. Untuk mendapatkan meta data, guna keperluan visualisasi hasil penelitian dari jurnal nasional khususnya jurnal sinta masih harus dilakukan secara manual. Peneliti harus copy dan paste satu persatu artikel yang dinginkan. Guna mempermudah penarikan meta data artikel yang termuat dalam jurnal sinta dengan sistem OJS maka dilakukan pengembangan aplikasi web scraping jurnal OJS. Metode web scraping yang dipilih adalah CSS Selector. Aplikasi dikembangkan dengan bahasa pemprograman python dan tambahan library BeautifulSoup, flask dan pandas. Dari hasil pengujian aplikasi scraping OJS mampu menarik meta data berupa data judul, abstrak, kata kunci, author, link. Kelemahan yang ditemukan tidak mampu menarik data pada jurnal dengan sistem OJS yang telah dilakukan perubahan interface web yang tidak standar OJS.

Kata Kunci: OJS; Web Scraping; CSS Selector; BeautifulSoup; Python

Abstract−Reference sources in research and scientific article writing are journals. For reputable international journals, a lot of software has been developed to retrieve meta data for research mapping. To obtain meta data, for the purposes of visualizing research results from national journals, especially Sinta's journal, it still has to be done manually. Researchers must copy and paste one by one the articles they want. In order to facilitate the retrieval of article meta data contained in a sinta journal with the OJS system, the development of an OJS journal scraping web application was carried out. The chosen web scraping method is the CSS Selector. The application is developed with the Python programming language and additional BeautifulSoup, flask and pandas libraries. From the results of testing the OJS scraping application was able to retrieve meta data in the form of title data, abstract, keywords, author, links. Weaknesses found were not being able to retrieve data in journals with the OJS system that had made changes to the non-OJS standard web interface.

Keywords: OJS; Web Scraping; CSS Selector; BeautifulSoup; Python

1. PENDAHULUAN

Web scraping adalah metode yang digunakan untuk mengekstrak informasi dari website. Web scraping telah menjadi solusi yang populer untuk mengekstrak metadata dari halaman web. Banyaknya data yang dipublikasikan melalui website membuat menarik para peneliti tertarik untuk mengumpulkan dan mengolah menjadi informasi.

Dari studi literatur didapatkan, web scraping banyak digunakan untuk dibidang ekonomi [1]–[5], media sosial [6]–[9]dan berita[10]. Belum banyak ditemukan yang membahas pemanfaatan web scraping dibidang penelitian. Banyak hasil penelitian dipublikasikan pada jurnal ilmiah. Di Indonesia tercatat terdapat 8.080 jurnal terakreditasi sinta [11]. Untuk mendapatkan gambaran umum tentang hasil penelitian yang termuat pada jurnal sinta, peneleliti kebanyakan membuka halaman journal dan membaca serta mencatat satu persatu dari informasi yang dibutuhkan. Hal ini akan memperlama untuk mendapatkan informasi terbaru tentang topik penelitian.

Web scraping dipandang sebagai sulusi yang tepat untuk mempercepat mendapatkan informasi penelitian terbaru[12]. Journal terakreditasi sinta mewajibkan menggunakan Open Journal Systems (OJS). OJS adalah platform sumber terbuka yang digunakan untuk mengelola jurnal akademik. OJS menyimpan metadata publikasi ilmiah, seperti judul, penulis, dan abstrak, dalam format terstruktur di dalam database. Informasi ini dapat digunakan untuk keperluan penelitian dan analisis.

Dalam konteks OJS, penggunaan web scraping dapat mempercepat proses pengumpulan data dan memungkinkan peneliti untuk memperoleh informasi yang relevan dengan cepat. Banyak metode web scraping digunakan, mulai dari hanya sekedar copy dan paste halaman web secara manual sampai penggunaan metode otomatis. Diantara metode web scraping yang banyak digunakan antara lain metode RegEx [13],[14], Html DOM[15],[16], Xpath [17],[18], CSS Selector[19]–[21].

Berdasarkan penelitian yang didapat menginformasikan bahwa RegEx memiliki penggunaan CPU dan memori yang paling sedikit dibandingkan dengan metode lainnya. Sedangkan Xpath membutuhkan waktu paling sedikit dibandingkan dengan metode lainnya. Metode CSS Selector paling kecil dalam hal penggunaan bandwidth dibandingkan dengan metode lainnya [22],[23].

Dari keunggulan yang ditawarkan dalam artikel ini menerapkan metode CSS Selector untuk pengumpulan data penelitian di OJS. Hal ini didasari karena OJS dibangun kombinasi css dan html yang memiliki struktur tetap dan CSS Selector menawarkan pemakaian bandwidth yang hemat yang akhirnya berelasi dengan biaya yang hemat. CSS Selector mengandalkan struktur DOM dari dokumen HTML, dan memungkinkan pengguna untuk memilih elemen berdasarkan nama tag, atribut, kelas, ID, atau hubungan dengan elemen lain. Setiap elemen pada halaman web dapat diidentifikasi dengan cara yang unik menggunakan kombinasi dari nama tag, atribut, kelas, ID, atau hubungan dengan elemen lain.

ISSN 2745-7966 (Media Online)
Vol 3, No 2, November 2022
Hal 63-68
https://djournals.com/resolusi

2. METODOLOGI PENELITIAN

2.1 CSS Selector

CSS Selector adalah teknik untuk memilih elemen pada halaman web dengan menggunakan selektor CSS. CSS Selector dapat digunakan untuk memilih elemen berdasarkan nama tag, kelas, ID, atribut, atau hubungan dengan elemen lain pada halaman web.
Berikut ini adalah beberapa jenis CSS Selector yang sering digunakan dalam web scraping:

1. Selector Nama Tag: digunakan untuk memilih elemen berdasarkan nama tag-nya, seperti p untuk memilih semua elemen paragraf pada halaman web.
2. Selector Kombinasi Tag: digunakan untuk memilih elemen dengan menggunakan kombinasi nama tag, seperti div p untuk memilih semua elemen paragraf yang berada dalam elemen div.
3. Selector ID: digunakan untuk memilih elemen berdasarkan ID-nya, seperti #header untuk memilih elemen dengan ID "header".
4. Selector Class: digunakan untuk memilih elemen berdasarkan kelas CSS-nya, seperti .intro untuk memilih semua elemen yang memiliki kelas "intro".
5. Selector Atribut: digunakan untuk memilih elemen berdasarkan nilai atribut, seperti [href] untuk memilih semua elemen yang memiliki atribut "href", atau [href="#"] untuk memilih semua elemen yang memiliki atribut "href" dengan nilai "#" .
6. Selector Pseudo-Class: digunakan untuk memilih elemen berdasarkan keadaan-nya, seperti :hover untuk memilih elemen saat mouse berada di atas-nya.

Dalam web scraping, CSS Selector sangat berguna untuk menyeleksi dan mengekstrak informasi yang diperlukan dari halaman web dengan mudah. Beberapa library Python yang digunakan dalam web scraping salah satunya adalah BeautifulSoup [21].

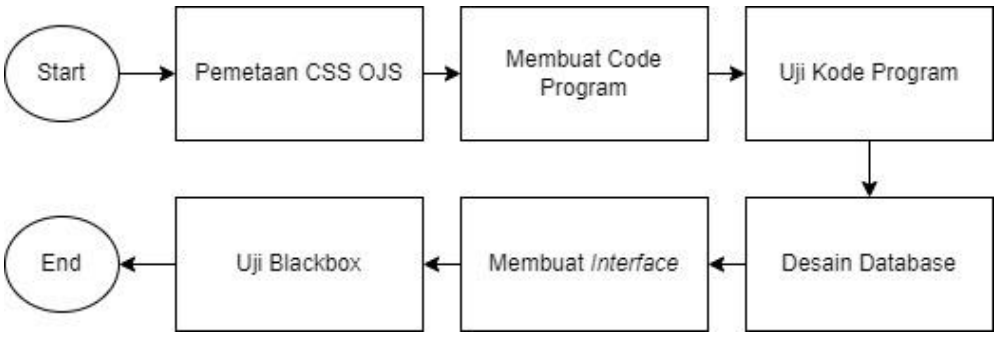
2.2 BeautifulSoup

BeautifulSoup adalah salah satu library Python yang paling populer dan berguna dalam web scraping. Library ini menyediakan API yang mudah digunakan untuk mengekstrak informasi dari halaman web dengan menggunakan bahasa pemrograman Python. BeautifulSoup menyediakan cara yang mudah untuk menavigasi dan mengekstrak data dari dokumen HTML atau XML. Dengan BeautifulSoup, pengguna dapat mengekstrak informasi dari halaman web dengan menggunakan teknik parsing dan manipulasi DOM.
Berikut ini adalah beberapa fitur yang disediakan oleh BeautifulSoup:

1. Parsing Dokumen HTML: BeautifulSoup menyediakan parser HTML yang powerful dan fleksibel untuk parsing dokumen HTML.
2. Navigasi Dokumen: BeautifulSoup menyediakan cara yang mudah untuk menavigasi dokumen HTML dengan menggunakan DOM.
3. Seleksi Elemen: BeautifulSoup menyediakan cara yang mudah untuk memilih elemen pada dokumen HTML dengan menggunakan CSS Selector atau regular expression.
4. Manipulasi Elemen: BeautifulSoup menyediakan cara yang mudah untuk memodifikasi elemen pada dokumen HTML, seperti menambahkan, menghapus, atau mengubah atribut.
5. Ekstraksi Data: BeautifulSoup menyediakan cara yang mudah untuk mengekstrak informasi dari dokumen HTML, seperti teks, atribut, link, atau gambar.

2.3 Tahapan penelitian

Tahapan penelitian ditunjukkan pada Gambar 1.



Gambar 1. Tahapan Penelitian

2.4 Pemetaan CSS OJS

Tahap ini untuk mengidentifikasi atribut meta itu berisi objek data. Kegiatan ini dilakukan untuk memperoleh informasi objek data yang akan diambil, seperti yang ditunjukkan pada Gambar 2. Data yang diambil meliputi Judul, author, abstrak, kata kunci, link.

ISSN 2745-7966 (Media Online)
Vol 3, No 2, November 2022
Hal 63-68
https://djournals.com/resolusi

```
<div id="articleTitle">
    <h1><b>Judul Artikel</b></h1>
</div>

<div id="articleAbstract">
    <h4>Abstrak</h4>
    <div>Isi Abstrak</div>
</div>

<div id="articleSubject">
    <h4>Keywords</h4>
    <br />
    <div>isi, kata kunci</div>
    <br />
</div>
```

Gambar 2. Meta data OJS

2.5 Membuat code program

Pada tahap ini dilakukan pembuatan kode program dengan bahasa python dan library BeautifulSoup berdasarkan meta data OJS. Secara umum kode program ditunjukan pada Gambar 3.

```
def procScraping():
        _linkvolumejurnal = request.form['linkvolumejurnal']
        # cek validasi apakah ada data dari post tidak
        if _linkvolumejurnal:
                def getLinks(url):
                        html_page = urlopen(url)
                        soup = BeautifulSoup(html_page)
                        links = []

                        for parent in soup.findAll("div", {"class": "tocTitle"}):
                                for link in parent.findAll('a',limit=1):
                                        links.append(link.get('href'))
                        return links
                judul = []
                abstrak1 = []
                katakunci1 = []
                for artikel in getLinks(_linkvolumejurnal):
                        html = urlopen(artikel).read()
                        soup = BeautifulSoup(html, 'html.parser')
                        judul_artikle = soup.find("div", {"id": "articleTitle"}).text
                        if judul_artikle == None:
                                continue

                        abstrak = soup.find("div", {"id": "articleAbstract"})
                        if abstrak == None:
                                continue
                        abstrak2 = abstrak.find("div").text

                        katakunci = soup.find("div",{"id": "articleSubject"})
                        if katakunci == None:
                                continue
                        katakunci2 = katakunci.find("div").text

                        judul.append(judul_artikle)
                        abstrak1.append(abstrak2)
                        katakunci1.append(katakunci2)

        # df =pd.DataFrame(data = numpyArray)
                df = pd.DataFrame({'Judul':judul,'Abstrak':abstrak1,'Kata Kunci':katakunci1,'Artikel': artikel})
                df.to_csv('jurnal.csv', index=False, mode='a', header=False) #For append it is 'a'.
                return redirect("http://localhost:5000/")
        else:
                return redirect("http://localhost:5000/")
```

Gambar 3. Kode program scraping OJS

2.6 Uji Kode Program

Dalam tahap ini dilakukan uji program apakah telah berhasil mengambil data dari jurnal yang menggunakan sistem OJS. Data masih disimpan dalam format csv.

2.7 Desain Database

Untuk menyimpan data yang berhasil di ekstrak dari jurnal OJS digunakan DBMS MYSQL dengan struktur data ditunjukan pada Gambar 4.

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
☐	1	id	int(11)			No	*None*		AUTO_INCREMENT	Change Drop More
☐	2	judul	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
☐	3	abstrak	text	utf8mb4_general_ci		No	*None*			Change Drop More
☐	4	katakunci	varchar(512)	utf8mb4_general_ci		No	*None*			Change Drop More
☐	5	author	varchar(500)	utf8mb4_general_ci		No	*None*			Change Drop More
☐	6	link	varchar(512)	utf8mb4_general_ci		No	*None*			Change Drop More

Gambar 4. Susunan Database

2.8 Kebutuhan fungsional

Kebutuhan fungsional aplikasi ditunjukan pada Tabel 1.

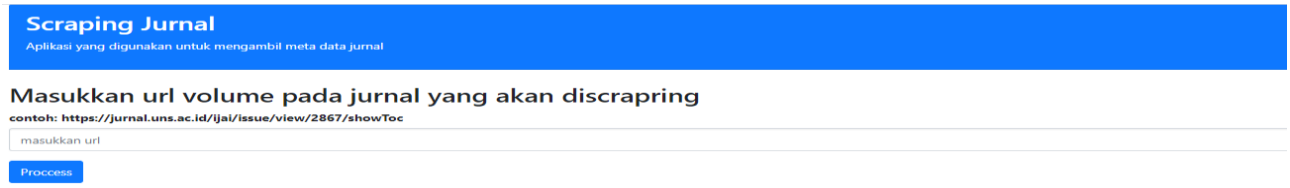
Tabel 1. Kebutuhan fungsional aplikasi scraping OJS

Kode FR	Deskripsi FR	Aktor	Dependensi
FR01	Form Input url Jurnal	Quest	
FR02	Proses tarik data artikel	Quest	
FR03	View data yang berhasil didapat	Quest	
FR04	Akses link yang tertera	Quest	
FR05	Pencarian data yang berhasil dikoleksi	Quest	
FR06	Export data dalam format .ris	Quest	

3.HASIL DAN PEMBAHASAN

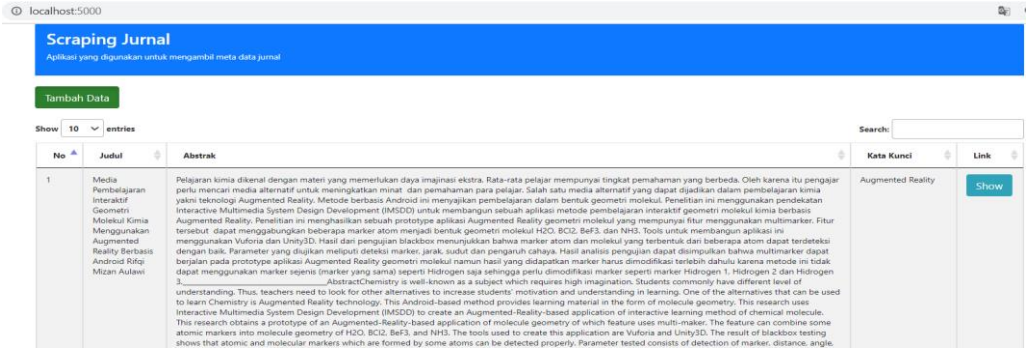
3.1 Proses Scraping OJS

Aplikasi scraping OJS dibuat dalam teknologi web dengan bahasa python dan tambahan library flask, pandas, BeautifulSoup. Aplikasi didesain tidak menggunakan sistem autentikasi. Untuk penggunaan cukup mengimputkan alamat url artikel yang dinginkan pada form seperti ditunjukan pada Gambar 4.



Gambar 4. Form input url

Ketika tombol proces diaktifkan maka kode program scraping yang dibuat dengan python dan libray BeautifulSoup akan bekerja, jika data berhasil didapatkan maka sistem akan menyimpannya kedalam database dan menampilkannya dalam bentuk tabel seperti pada Gambar 5.



Gambar 5. Hasil Proses Scraping

3.2 Pengujian blackbox

Untuk mengetahui berjalannya semua fungsional dilakukan uji blackbox. Nama jurnal dengan sistem OJS yang digunakan untuk uji coba meliputi jurnal IJAI, JURIKOM, Attarbiyah, Hipotenusa, Ijoresh, IJIP, Ijtihad, Resolusi. Hasil pengujian blackbox ditunjukan pada Tabel 2.

ISSN 2745-7966 (Media Online)
Vol 3, No 2, November 2022
Hal 63-68
https://djournals.com/resolusi

Tabel 4. Uji blackbox aplikasi scraping OJS

No	Fungsionalitas	Skenario uji	Hasil yang diharapkan	Keterangan
1	Form Input url Jurnal	Quest input alamat url jurnal yang ditentukan	Sistem mampu menangkap dan membaca alamat url yang inputkan	Valid
2	Proses tarik data artikel	Quest menekan tombol proses	Sistem akan menarik data berupa judul , abstrak, kata kunci, author, link tanpada sistem berhenti atau muncul pesan error	Valid
6	View data yang berhasil didapat	Quest aksess tabel data	Sistem akan memunculkan tabel yang berisi data judul , abstrak, kata kunci, author, link	Valid
7	Akses link yang tertera	Quest aksess link jurnal	Sistem akan memberikan alamat link yang tepat, dan link berhasil dibuka dengan sempurna	Valid
8	Pencarian data yang berhasil dikoleksi	Quest melakukan pencarain data	Sistem akan memproses kata kunci pencarian dan menampilkan data yang relevan.	Valid
10	Export data dalam format .ris	Quest export data	Sistem akan mengemas data dalam format .ris	Valid

Aplikasi scraping ojs hanya berhasil untuk menarik data pada sistem jurnal yang menggunakan desain interface standar OJS, sistem jurnal yang telah terjadi modifikasi CSS yang tidak standar tidak berhasil untuk diproses.

4. KESIMPULAN

Berdasarkan pengujian dengan metode blackbox maka dapat disimpulkan bahwa penggunaan metode CSS Selector dan library BeautifulSoup mampu menarik data jurnal judul, abstrak, kata kunci, author dengan sistem OJS dan mampu memformat data dalam format .ris untuk keperluan pengolahan data selanjutnya. Kelemahan yang didapat adalah tidak mampu menarik data pada jurnal dengan sistem OJS yang mana struktur interface webnya telah keluar dari standar.

UCAPAN TERIMAKASIH

Terima kasih disampaikan kepada IAIN Salatiga dan semua pihak yang telah mendukung terlaksananya penelitian ini.

REFERENCES

[1] J. Hillen, “Web scraping for food price research,” Br. Food J., vol. 121, no. 12, 2019, doi: 10.1108/BFJ-02-2019-0081.
[2] F. Djiwadikusumah, G. H. Irawan, and R. Haekal Al-Fadilah, “Web scraping situs e-commerce menggunakan teknik parsing dom,” J. Siliwangi, vol. 7, no. 2, 2021.
[3] D. F. Setiawan, T. Tristiyanto, and A. Hijriani, “APLIKASI WEB SCRAPING DESKRIPSI PRODUK,” J. Teknoinfo, vol. 14, no. 1, 2020, doi: 10.33365/jti.v14i1.498.
[4] D. D. A. Yani, H. S. Pratiwi, and H. Muhardi, “Implementasi Web Scraping untuk Pengambilan Data pada Situs Marketplace,” J. Sist. dan Teknol. Inf., vol. 7, no. 4, 2019, doi: 10.26418/justin.v7i4.30930.
[5] M. Djufri, “PENERAPAN TEKNIK WEB SCRAPING UNTUK PENGGALIAN POTENSI PAJAK (STUDI KASUS PADA ONLINE MARKET PLACE TOKOPEDIA, SHOPEE DAN BUKALAPAK),” J. BPPK Badan Pendidik. dan Pelatih. Keuang., vol. 13, no. 2, 2020, doi: 10.48108/jurnalbppk.v13i2.636.
[6] R. Baskara and F. Rahma, “Implementasi Web Scraping Pada Media Sosial Instagram,” Automata, vol. 3, pp. 1–3, 2022.
[7] R. Crystal Pereira and T. Vanitha, “Web Scraping of Social Networks,” Int. J. Innov. Res. Comput. Commun. Eng. (An ISO, vol. 3297, no. 7, 2015.
[8] I. Dongo, Y. Cadinale, A. Aguilera, F. Martínez, Y. Quintero, and S. Barrios, “Web Scraping versus Twitter API: A Comparison for a Credibility Analysis,” 2020, doi: 10.1145/3428757.3429104.
[9] M. I. Habibie, T. Widiaputra, and Y. Yulianingsani, “WEB SCRAPING OF DISEASE INFORMATION FROM SOCIAL MEDIA TWITTER,” J. Teknoinfo, vol. 16, no. 2, 2022, doi: 10.33365/jti.v16i2.1871.
[10] S. Satriajati, S. B. Panuntun, and S. Pramana, “IMPLEMENTASI WEB SCRAPING DALAM PENGUMPULAN BERITA KRIMINAL PADA MASA PANDEMI COVID-19,” Semin. Nas. Off. Stat., vol. 2020, no. 1, 2021, doi: 10.34123/semnasoffstat.v2020i1.578.
[11] sinta, “journals,” kemendikbud, 2023. https://sinta.kemdikbud.go.id/journals.
[12] S. E. Chasins, M. Mueller, and R. Bodik, “Rousillon: Scraping distributed hierarchical web data,” 2018, doi: 10.1145/3242587.3242661.
[13] I. Onyenwe, S. Ogbonna, E. Onyedimma, O. Ikechukwu-Onyenwe, and C. Nwafor, “Developing Smart Web-Search using Regex,” Int. J. Nat. Lang. Comput., vol. 11, no. 3, 2022, doi: 10.5121/ijnlc.2022.11303.
[14] Z. Wu, B. Ericson, and C. Brooks, “Regex Parsons: Using Horizontal Parsons Problems to Scaffold Learning Regex,” 2021, doi: 10.1145/3488042.3489968.
[15] A. Rahmatulloh and R. Gunawan, “Web Scraping with HTML DOM Method for Data Collection of Scientific Articles from Google Scholar,” Indones. J. Inf. Syst., vol. 2, no. 2, 2020, doi: 10.24002/ijis.v2i2.3029.

[16] V. Mitra, H. Sujaini, and A. B. P. Negara, "Rancang Bangun Aplikasi Web Scraping untuk Korpus Paralel Indonesia - Inggris dengan Metode HTML DOM," J. Sist. dan Teknol. Inf., vol. 5, no. 1, 2017.
[17] P. Gao, H. Han, J. Guo, and M. Saeki, "Stable web scraping: An approach based on neighbour zone and path similarity of page elements," Int. J. Web Eng. Technol., vol. 13, no. 4, 2018, doi: 10.1504/IJWET.2018.097561.
[18] R. Yaqoob, Sanaa, M. Haris, Samadyar, and M. A. Shah, "The Price Scraping Bot Threat on E-commerce Store Using Custom XPATH Technique," 2021, doi: 10.23919/ICAC50006.2021.9594223.
[19] M. S. Rohman, H. A. Santoso, G. W. Saraswati, N. Anisa, and S. Winarsih, "Pemanfaatan Topic-Focused Crawler untuk Pembangunan Corpus Berita Bencana menggunakan Teknik Scrapy CSS Selector," Semin. Nas. APTIKOM, 2019.
[20] E. Uzun, "A regular expression generator based on CSS selectors for efficient extraction from HTML pages," Turkish J. Electr. Eng. Comput. Sci., vol. 28, no. 6, 2020, doi: 10.3906/ELK-2004-67.
[21] J. Attardi, "CSS Selectors," in Modern CSS, 2020.
[22] R. Gunawan, A. Rahmatulloh, I. Darmawan, and F. Firdaus, "Comparison of Web Scraping Techniques : Regular Expression, HTML DOM and Xpath," 2019, doi: 10.2991/icoiese-18.2019.50.
[23] I. Darmawan, M. Maulana, R. Gunawan, and N. Widiyasono, "Evaluating Web Scraping Performance Using XPath, CSS Selector, Regular Expression, and HTML DOM With Multiprocessing Technical Applications," Int. J. Informatics Vis., vol. 6, no. 4, 2022, doi: 10.30630/joiv.6.4.1525.