BAB II
TINJAUAN PUSTAKA

2.1 Penelitian Terkait

Penelitian yang bertujuan untuk membangun *website* yang dibutuhkan oleh berbagai pengguna *website E-Commerce* agar mempermudah dalam melakukan pencarian produk dengan menggunakan teknik *web scraping* dari *website* Shopee, Tokopedia, Lazada, Blibli, Bukalapak. Sudah banyak penelitian yang dilakukan mengenai rancang bangun *website* yang dapat melakukan *web scraping* pada objek yang berbeda. Berikut adalah penelitian-penelitian yang sebelumnya sudah dilaksanakan dan berkaitan dengan penelitian yang diangkat oleh penulis.

Dalam penelitian yang berjudul "Implementasi *Web scraping* untuk Pengambilan Data pada Situs *Marketplace*" [12]. Memiliki masalah berupa pengguna *E-Commerce* kesulitan dalam memperoleh produk yang diinginkan dengan hasil penjualan terbaik, melibatkan perbandingan harga di antara situs dan produk, serta meninjau ulasan dari pembeli. *Website* berhasil dibuat dengan menerapan teknik *web scraping* pada situs Bukalapak, Elevenia, dan JD.id serta telah diuji menggunakan *white box testing* dan *black box testing* dan memberikan hasil analisis bahwa *website* berjalan sesuai fungsionalitas dan menampilkan produk terbaik dari gabungan hasil pencarian di tiga situs web *marketplace* sesuai kata kunci yang dimasukkan oleh pengguna.

Dalam penelitian yang berjudul "*Web scraping* Situs *E-Commerce* Menggunakan Teknik Parsing DOM" [10]. Terjadi kendala di kalangan pengguna *E-Commerce* dalam mencari produk yang diinginkan, memperoleh hasil penjualan terbaik, dan melakukan perbandingan harga antar situs perdagangan daring. Dalam mengatasi permasalahan ini, dilakukan rancang bangun *website* dengan menerapan teknik *web scraping* pada platform-platform seperti Tokopedia, Shopee, JD.in, Elevenia, Blibli, dan Bukalapak. *Website* berhasil

dibuat dan penerapkan *web scraping* berhasil dilakukan serta memberikan hasil yang sesuai dengan harapan awal dari tiga situs *E-Commerce*.

Dalam penelitian yang berjudul "Aplikasi *Web scraping* Deskripsi Produk" yang melakukan teknik serupa [13]. Memiliki masalah berupa deskripsi produk memiliki jumlah kata yang sangat banyak dan beragam sehingga dapat menghabiskan waktu yang sangat lama dalam mengelola produk. Penelitian ini membangunan sebuah *website* menggunakan *framework Laravel* dengan metode *waterfall* dan menerapan *web scraping* parsing HTML. Hasilnya, sistem informasi *web scraping* berbasis web dengan menggunakan *framework Laravel* 5.7.21 berhasil dilakukan dan telah diuji menggunakan *black box testing* serta berhasil melakukan *scraping* data produk pada 9 *E-Commerce* yaitu Alibaba, Amazon, Blanja.com, Bukalapak, Kriya, Lazada, Tokopedia, Zalora, dan Zilingo.

Dalam penelitian yang berjudul "Penerapan Teknik *Web scraping* Untuk Pencarian Produk Terlaris Di Berbagai Situs *E-Commerce* Indonesia" [14]. Memiliki masalah berupa para pebisnis bingung dalam memilih barang yang akan dijual atau dibeli dengan alasan bahwa saat melakukan transaksi, terkadang pebisnis tidak tahu apa barang yang sedang laris dijual pasar. *Website* berhasil dibuat dengan menampilkan data produk terlaris beserta detail produk yang sesuai dengan yang ada pada *E-Commerce* serta menerapan *web scraping* pada *website* Shopee dan Lazada dan telah diuji menggunakan *black box testing* yang memberikan hasil sesuai dengan kriteria kebutuhan pengambilan data yang difokuskan.

Dalam penelitian yang berjudul "Pemanfaatan Teknik *Web scraping* Python Untuk Sistem Pencarian Produk Di Toko *Online*" [15]. Muncul kesulitan dalam mencari barang melalui mesin pencari seperti Google dan Yahoo, karena hasilnya berupa daftar alamat situs *web*. Hal ini memaksa masyarakat untuk mengunjungi satu per satu alamat toko *online* guna mencari dan membandingkan barang. Dalam mengatasi permasalahan ini, dilakukan pengembangan aplikasi berbasis *website* yang menerapkan *web scraping* pada platform Tokopedia, Shopee, Elevenia, dan Blibli. Sistem pencarian produk

dengan teknik *web scraping* berhasil dilakukan dan telah diuji menggunakan *black box testing*, Hasilnya mempermudah para *customer* dalam mencari barang dan mengisi deskripsi produk dengan mudah dan cepat.

Dalam penelitian yang berjudul "Implementasi *Web scraping* Untuk Mengumpulkan Informasi Produk Dari Situs *E-Commerce* Dan *Marketplace* Dengan Teknik Pemrosesan Paralel" [16]. Ditemui kendala di mana pembeli harus mengeluarkan waktu yang cukup lama untuk mencari barang sesuai dengan kriteria yang diinginkan. Dalam mengatasi hal ini, diterapkan teknik *web scraping* dengan berbagai jumlah *thread*. Hasilnya, proses *scraping* dilakukan secara paralel dengan *multithreading*, mempercepat proses pengumpulan informasi produk dari beberapa situs secara bersamaan. Teknik ini telah diuji menggunakan *black box testing*.

Dalam penelitian yang berjudul "Analisis Data Produk Elektronik Di *E-Commerce* Dengan Metode Algoritma K-Means Menggunakan *Python*" [17]. Memiliki masalah berupa ulasan produk yang diberikan pelanggan akan berpengaruh terhadap penjualan di *E-Commerce*, maka dari itu dilakukan *scraping* data komentar dari produk. Penerapan teknik *web scraping* pada kolom komentar produk berhasil dilakukan dan memberikan hasil *Word cloud graphics* didapatkan kata kata yang dominan dari ulasan suatu produk.

Berdasarkan penilitian-penelitian sebelumnya yang telah disebutkan, penulis menganalisis bahwa dari metode *web scraping* parsing HTML yang diterapkan pada sebuah *website* sudah memiliki hasil sesuai serta pengujian dengan *black box testing* berjalan seperti yang diharapkan. Oleh karena itu pada penelitian ini, penulis menggunakan metode *web scraping* parsing HTML untuk diterapkan pada sistem *scraping* produk dan menggunakan *black box testing* sebagai pengujian fungsional. Ringkasan pada penelitan sebelumnya dapat dilihat pada Tabel 2.1.

Tabel 2.1 Ringkasan penelitian sebelumnya

NO	Judul	Penulis	Rumusan Masalah	Metode	Perbedaan	Persamaan	Hasil
1.	Implementasi *Web scraping* untuk Pengambilan Data pada Situs *Marketplace*. 2019 [12].	Dhita Deviacita Ayani , Helen Sasty Pratiwi , Hafiz Muhardi.	Pengguna *E-Commerce* kesulitan dalam memperoleh produk yang diinginkan dengan hasil penjualan terbaik, melibatkan perbandingan harga di antara situs dan produk, serta meninjau ulasan dari pembeli.	Parsing DOM, *black box testing*, *white box testing*.	Objek *website* menggunakan Elevenia dan JD.id	Melakukan *web scraping* dari *website* Bukalapak	*Website* berhasil dibuat dengan menerapan teknik *web scraping* serta telah diuji menggunakan *white box testing* dan *black box testing* dan memberikan hasilv bahwa *website* berjalan sesuai fungsionalitas dan menampilkan produk terbaik dari gabungan hasil pencarian di tiga situs *E-Commerce* sesuai kata kunci yang dimasukkan oleh pengguna.
2.	*Web scraping* Situs *E-Commerce*	Farhan Djiwadikusumah, Genta Hayindra	Untuk mencari produk yang diinginkan,	Parsing DOM	Menggunakan bahasa pemrograman	Menggunakan metode Parsing HTML untuk	*Website* berhasil dibuat dan penerapkan *web*

NO	Judul	Penulis	Rumusan Masalah	Metode	Perbedaan	Persamaan	Hasil
	Menggunakan Teknik Parsing DOM. 2021 [10].	Irawan, Rifqy Haekal al-Fadilah	dengan hasil penjualan terbaik, serta perbandingan harga produk antar situs jual beli komersial membutuhkan waktu yang lama.		JavaScript dan library Parsing DOM untuk melakukan *web scraping* dan menggunakan *website* JD.in dan elevenia	melakukan *web scraping* dan menggunakan situs Shopee, Tokopedia, Bukalapak	*scraping* berhasil dilakukan serta memberikan hasil yang sesuai dengan harapan awal dari tiga situs *E-Commerce*.
3.	Aplikasi *Web scraping* Deskripsi Produk. 2020 [13].	Dana Febri Setiawan, Tristiyanto, Astria Hijriani.	Setiap produk memiliki deskripsi dan harga yang berbeda, pengisian deskripsi dengan jumlah kata yang sangat banyak dan beragam dapat menghabiskan waktu yang sangat lama dalam	Parsing HTML, *waterfall*, *black box testing*	Menggunakan *website* Zalora, Blanja.com, Kriya.co.id, Zilingo, Amazon, dan Alibaba	Melakukan *web scraping* dengan metode Parsing HTML dari *website* Tokopedia, Bukalapak, Lazada	Sistem informasi *web scraping* berbasis *web* dengan menggunakan *framework Laravel 5.7.21* berhasil dilakukan dan telah diuji menggunakan *black box testing* serta berhasil melakukan *scraping* data produk dengan metode parsing

NO	Judul	Penulis	Rumusan Masalah	Metode	Perbedaan	Persamaan	Hasil
			mengelola produk.				HTML pada 9 *E-Commerce.*
4.	Penerapan Teknik *Web scraping* Untuk Pencarian Produk Terlaris Di Berbagai Situs *E-Commerce* Indonesia. 2022 [14].	Rais Saputra , Faradilla Laksmita Devi , Asep Supriyanto , Putri Anggun Sari	dalam memilih produk yang ingin dijual, masyarakat bingung untuk menentukan produk apa yang dibutuhkan karena ketidaktahuan produk apa yang sedang laris di pasaran	Rapid Application Development (RAD), selenium, *black box testing*.	Menggunakan metode Rapid Application Development (RAD)	*Web scraping* menggunakan bahasa pemrograman python dan mengambil dari *website* Shopee dan Lazada	*Website* berhasil dibuat dengan menampilkan data produk terlaris beserta detail produk yang sesuai dengan yang ada pada *E-Commerce* Lazada dan Shopee. Perancangan dan hasil telah sesuai dilakukan dengan pengujian *black box testing* yang memberikan hasil sesuai dengan kriteria kebutuhan pengambilan data yang difokuskan.
5.	Pemanfaatan Teknik *Web scraping* Python Untuk Sistem	Adi Sopian, Andy Dharmalau, Alpindo	Dalam mencari barang melalui mesin pencari seperti Google	Observasi, *black box testing*	Mengambil data *scraping* dari *website* Elevania,	Melakukan *web scraping* dengan bahasa	Pengembangan aplikasi berbasis *website* yang menerapkan *web*

NO	Judul	Penulis	Rumusan Masalah	Metode	Perbedaan	Persamaan	Hasil
	Pencarian Produk Di Toko *Online*. 2022 [15].		dan Yahoo, hasilnya berupa daftar alamat situs *web*. Hal ini memaksa masyarakat untuk mengunjungi satu per satu alamat toko *online* guna mencari dan membandingkan barang.		menggunakan halaman login	pemrograman python	*scraping* berhasil dibuat dan telah diuji menggunakan *black box testing*, Hasilnya mempermudah para customer dalam mencari barang dan mengisi deskripsi produk dengan mudah dan cepat.
6.	Implementasi *Web scraping* Untuk Mengumpulkan Informasi Produk Dari Situs *E-Commerce* Dan Marketplace Dengan Teknik Pemrosesan	Albert Stevan Yondra , Dedi Triyanto , Syamsul Bahri	Setiap situs *E-Commerce* dan marketplace menawarkan produk berbeda-beda, Ini menyebabkan pembeli harus mengunjungi tiap situs untuk mencari dan membandingkan	Studi Literatur, Selenium, *black box testing*.	Menggunakan metode pemrosesan paralel, menggunakan python library selenium	Menggunakan bahasa pemrograman python untuk melakukan *web scraping*	Hasil pengujian dengan jumlah thread yang berbeda-beda, didapatkan proses *scraping* yang dilakukan secara paralel dengan multithreading dapat mempercepat proses

NO	Judul	Penulis	Rumusan Masalah	Metode	Perbedaan	Persamaan	Hasil
	Paralel. 2022 [16].		produk dari tiap situs untuk menentukan pilihan terbaik.				pengumpulan informasi produk dari beberapa situs sekaligus.
7.	Analisis Data Produk Elektronik Di *E-Commerce* Dengan Metode Algoritma K-Means Menggunakan Python. 2021 [17].	Ainur Rahman dan Heri Suroyo	Pada suatu produk yang memiliki banyak ulasan, membutuhkan waktu yang lama untuk melihat secara satu per satu.	K-Means	Menggunakan metode K-Means	Melakukan *web scraping* dengan bahasa pemrograman python	*Scraping* pada ulasan produk berhasil dilakukan dan digambarkan melalui *wordcloud*.

2.2 Dasar Teori

2.2.1 *E-Commerce*

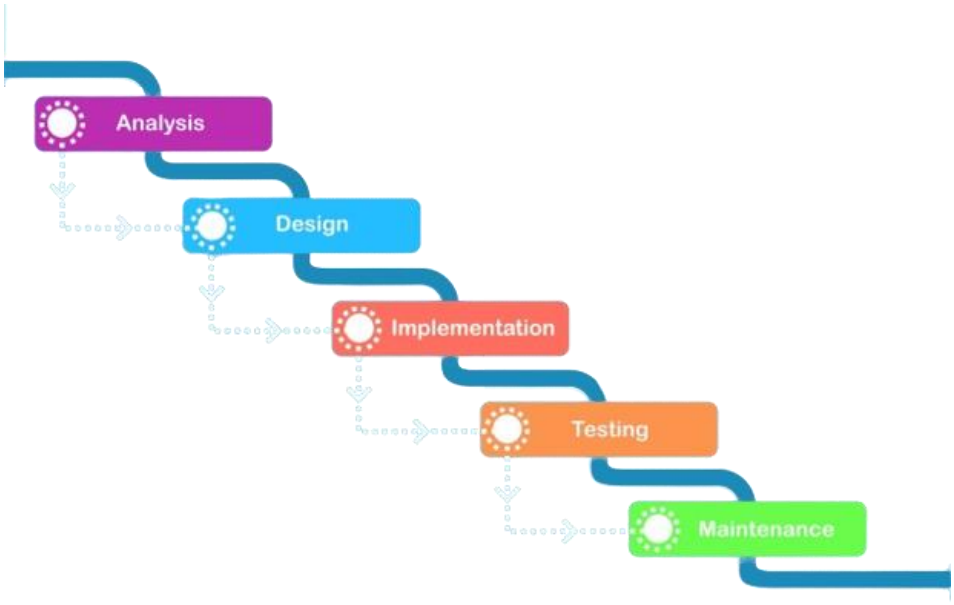
E-Commerce merupakan kegiatan transaksi jual-beli yang terjadi secara daring melalui jaringan komputer atau internet. Model-model *E-Commerce* mencakup *business to business (B2B), business to consumer (B2C), business to business to consumer (B2B2C), consumer* to *business* (C2B), dan *Collaborative Commerce*. Semua transaksi *E-Commerce* dilakukan secara *online* melalui platform *website.* [18].

E-Commerce telah mengubah cara orang berbelanja dan berbisnis, menghilangkan keterbatasan geografis dan waktu. Konsumen dapat menjelajahi berbagai produk dari berbagai penjual di seluruh dunia, membandingkan harga, membaca ulasan, dan melakukan pembelian dengan mudah. Sedangkan penjual dapat menjangkau pelanggan potensial di berbagai lokasi, mengurangi biaya *overhead* fisik, dan mengoptimalkan proses penjualan.

E-Commerce memiliki sejumlah keuntungan, seperti kenyamanan, aksesibilitas global, pilihan produk yang luas, dan efisiensi transaksi. Namun, ada juga tantangan yang terkait dengan *E-Commerce*, seperti keamanan data, kepercayaan konsumen, dan persaingan yang intensif.

2.2.2 Metode *Waterfall*

Metode *waterfall* adalah salah satu model atau pendekatan yang umum digunakan dalam *Software Development Life Cycle* (SDLC) . Model ini dikenal sebagai "*waterfall*" karena proses pengembangan mengalir dalam tahapan yang berurutan, seperti aliran air terjun. Gambar dari SDLC *waterfall* dapat dilihat pada Gambar 2.1 berikut.



Gambar 2.1 Tahapan-tahapan sdlc *waterfall*

Dalam metode *Waterfall*, setiap tahap pengembangan perangkat lunak dilakukan secara berurutan dan harus selesai sebelum memasuki tahap berikutnya. Tahapan-tahapan tersebut meliputi [19]:

1. Perencanaan analisis: Pada tahap ini, pengembang menentukan apa saja yang dibutuhkan di dalam aplikasi yang akan dikerjakan. Bagaimana cara membuat *website* yang menampilkan hasil *web scraping* produk dari berbagai situs marketplace.
2. Desain : Tahap ini, pengembang mulai merancang kerangka aplikasi yang bertujuan untuk memberikan gambaran mengenai tampilan aplikasi. Tahap ini dilakukan dengan membuat desain *use case* diagram dan desain *interface* sistem.
3. Implementasi: Tahap ini pengembang melakukan pengkodean program. Membuat sebuah *website* yang dapat melakukan *web scraping* dengan metode *parsing HTML* yang akan mengambil deskripsi produk kemudian akan dimasukkan otomatis kedalam database.
4. Pengujian: Tahap ini pengembang melakukan pengujian terhadap aplikasi yang telah dibuat apakah output sesuai dengan apa yang

pengembang harapkan. Dengan menggunakan *black box testing* untuk mengetahui fungsionalitas dari *scraping* yang telah dilakukan.

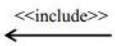
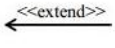
5. Pemeliharaan: Pada tahap ini, pengembang melakukan perbaikan berkala kepada program apabila terdapat *bugs* atau *error* yang terjadi setelah peluncuran aplikasi.

2.2.3 UML

UML (*Unified Modelling Language*) adalah salah satu alat bantu yang sangat handal di dunia pengembangan sistem berorientasi obyek yang telah menjadi standar dalam industri untuk visualisasi dalam merancang dan mendokumentasikan sistem piranti lunak [20]. UML juga disebut bahasa standar yang digunakan untuk menggambarkan, merancang, dan mendokumentasikan sistem perangkat lunak. Dengan menggunakan UML, para pengembang perangkat lunak dapat berkomunikasi dengan jelas dan memodelkan sistem secara visual, memfasilitasi analisis yang mendalam, perancangan yang sistematis, dan dokumentasi yang konsisten dari sistem perangkat lunak yang kompleks.

2.2.4 *Use Case Diagram*

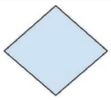
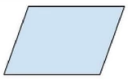
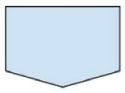
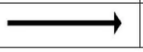
Use case merupakan langkah pertama dalam memodelkan sebuah sistem. *Use case* merupakan pemodelan untuk kebutuhan sebuah sistem fungsional, setiap *use case* digambarkan sebagai kunci dari suatu skenario yang dilakukan oleh aktor dan diringkas dalam sebuah batas sistem, setiap *use case* dihubungkan dengan sebuah garis notasi [21]. *Use case diagram* membantu dalam pemodelan fungsionalitas sistem dengan mengidentifikasi aktor yang terlibat, tindakan atau proses yang dilakukan oleh aktor, serta hubungan antara aktor dan *use case* yang menjelaskan interaksi sistem dengan pengguna. *Use case diagram* memberikan gambaran yang jelas tentang kebutuhan pengguna, fungsi yang diharapkan, dan batasan sistem yang sedang dikembangkan, sehingga mempermudah pemahaman dan komunikasi antara pemangku kepentingan dalam pengembangan perangkat lunak.

Simbol	Keterangan
	Aktor : Mewakili peran orang, sistem yang lain, atau alat ketika berkomunikasi dengan *use case*
	*Use case* : Abstraksi dan interaksi antara sistem dan aktor
	Association : Abstraksi dari penghubung antara aktor dengan use case
	Generalisasi : Menunjukkan spesialisasi aktor untuk dapat berpartisipasi dengan use case
	Menunjukkan bahwa suatu use case seluruhnya merupakan fungsionalitas dari use case lainnya
	Menunjukkan bahwa suatu use case merupakan tambahan fungsional dari use case lainnya jika suatu kondisi terpenuhi

Gambar 2.2 Simbol-simbol pada *use case diagram* [22]

2.2.5 *Flowchart*

Flowchart adalah langkah-langkah pemecahan masalah yang ditulis atau dilambangkan dengan simbol-simbol tertentu [23]. *Flowchart* adalah representasi visual dari urutan langkah-langkah atau alur logika dalam suatu proses. *Flowchart* digunakan untuk menggambarkan pemrosesan informasi, pengambilan keputusan, atau algoritma dengan menggunakan simbol-simbol grafis seperti kotak, panah, dan berlian. Setiap simbol dalam *flowchart* mewakili tindakan atau keputusan tertentu, dan panah menghubungkan simbol-simbol tersebut untuk menunjukkan aliran urutan yang tepat. *Flowchart* membantu dalam pemahaman dan komunikasi yang jelas mengenai langkah-langkah suatu proses, memperjelas alur logika, mengidentifikasi masalah, serta membantu dalam perancangan, analisis, dan dokumentasi sistem perangkat lunak atau proses bisnis.

No.	Simbol Flowchart	Nama	Arti Simbol Flowchart
1		*Terminator*	Awal atau akhir konsep (prosedur)
2		*Process*	Proses operasional
3		*Document*	Dokumen atau laporan berupa *print out*
4		*Decision*	Keputusan atau sub-point. Garis yang terhubung dengan bentuk *decision* merujuk pada situasi-situasi yang berbeda sesuai dengan keputusan yang digambarkan
5		Data	Input dan Output (Contohnya, Input: feedback dari pelanggan. Output: desain produk baru)
6		*On-Page Reference/ Connector*	Penghubung alur dalam halaman yang sama
7		*Off-Page Reference/ Off-Page Connector*	Penghubung alur dalam halaman yang berbeda
8		*Flow*	Arah alur dalam konsep (prosedur)

Gambar 2.3 Simbol-simbol pada *flowchart* [24]

2.2.6 SQLite

SQLite adalah pustaka *open source* bahasa pemrograman C yang menyediakan mesin basis data relasional yang dapat dioperasikan dengan bahasa kueri SQL. SQLite adalah mesin database relasional ringan yang dapat diintegrasikan langsung ke dalam suatu aplikasi [25]. SQLite dirancang sebagai database berbasis file tunggal, yang berarti seluruh database disimpan dalam satu file, hal itu menyebabkan SQLite cocok untuk aplikasi yang membutuhkan basis data lokal dengan akses langsung ke file. SQLite tidak memerlukan proses instalasi atau konfigurasi, berbeda dengan MySQL dan PosgreSQL yang memerlukan instalasi serta konfigurasi terpisah.

2.2.7 Flask

Flask merupakan sebuah kerangka kerja yang menggunakan bahasa pemrograman Python. Flask termasuk kedalam *microframework* karena tidak memerlukan suatu alat atau pustaka tertentu dalam penggunaannya [26]. Flask

dirancang untuk membuat pengembangan aplikasi web dengan Python menjadi lebih mudah dan sederhana. Meskipun bersifat mikro, Flask tetap sangat fleksibel dan dapat diintegrasikan dengan berbagai ekstensi untuk memperluas fungsionalitasnya.

2.2.8 *Web scraping*

Web scraping adalah teknik pengumpulan data yang melibatkan ekstraksi informasi secara otomatis dari halaman web [12]. *Web scraping* dapat dilakukan dengan menggunakan bahasa pemrograman *Python, JavaScript, PHP,* dll. Dengan menggunakan program atau script khusus, *web scraping* memungkinkan pengambilan data dari berbagai sumber yang ada di internet, termasuk situs web, forum, media sosial, dsb. Teknik ini memanfaatkan struktur HTML atau API (*Application Programming Interface*) untuk mengakses dan mengambil data yang diperlukan. *Web scraping* dapat dilakukan dengan berbagai metode, seperti mengambil teks, gambar, tautan, atau data terstruktur lainnya.

2.2.8.1 Jenis *web scraping*

Berikut adalah jenis-jenis *web scraping* yang umum digunakan:

1. *Web scraping* Berbasis HTML Parsing. Jenis *web scraping* ini melibatkan pemecahan struktur HTML menggunakan pustaka seperti *BeautifulSoup* [27]. Pada dasarnya, ini adalah metode untuk memparsing dokumen HTML dan mengekstrak data yang diinginkan dari elemen-elemen tertentu.
2. *Web scraping* Berbasis API. Beberapa situs web menyediakan API (*Application Programming Interface*) yang memungkinkan pengaksesan data mereka secara terstruktur dan terdokumentasi [28]. Dalam hal ini, *web scraping* dilakukan dengan mengakses *endpoint* API tersebut dan mengambil data yang diinginkan.
3. *Web scraping* Berbasis Pengendali Browser. Metode ini melibatkan penggunaan otomatisasi peramban web untuk mengakses dan mengekstrak data dari halaman web [29]. Pustaka seperti *Selenium* atau

Puppeteer dapat digunakan untuk mengendalikan peramban web dan melakukan tindakan seperti mengisi formulir, mengklik tombol, dan mengekstrak data dari halaman yang dimuat.

4. *Web scraping* Berbasis *Scrapy*. *Scrapy* adalah sebuah *framework web scraping* berbasis *Python* yang memudahkan proses pengambilan data dari berbagai situs web. Dengan *Scrapy*, pengembang dapat mengkonfigurasi *spider* atau *bot* web yang dapat mengunjungi halaman-halaman web, mengekstrak data yang diinginkan, dan mengikuti tautan untuk menjelajahi lebih jauh situs web yang memerlukan *login* atau berurusan dengan *cookie* [30].

2.2.8.2 Manfaat *web scraping*

Web scraping memungkinkan pengembang untuk mengumpulkan data secara otomatis dari berbagai situs web. Pengembang dapat mengumpulkan data seperti harga produk, ulasan pengguna, informasi kontak, informasi pasar, dsb. Data ini dapat digunakan untuk analisis, riset pasar, pengambilan keputusan, pengembangan produk, dan tujuan lainnya [28].

Web scraping juga dapat digunakan untuk mengumpulkan informasi publik yang terdapat di situs web pemerintah, situs berita, forum, atau sumber-sumber *online* lainnya [28]. Seperti pengumpulan data demografi, informasi cuaca, jadwal acara, atau informasi publik lainnya yang dapat digunakan untuk keperluan analisis, perencanaan, atau pengambilan keputusan.

Data yang diambil melalui *web scraping* dapat digunakan untuk mengembangkan aplikasi atau layanan baru. Seperti membangun situs berita, membangun sistem rekomendasi berbasis konten, membangun platform pembanding harga, atau membangun aplikasi berbasis data lainnya.

2.2.8.3 Kendala *web scraping*

Banyak situs web memiliki kebijakan yang melarang atau membatasi akses otomatis dan pengambilan data mereka. Beberapa situs web mungkin menggunakan mekanisme seperti pembatasan akses IP, verifikasi CAPTCHA, atau penghalang teknis lainnya untuk mencegah *web scraping* [31].

Beberapa situs juga terkadang dapat merubah struktur halaman web mereka. Hal ini dapat menyebabkan kesulitan dalam mempertahankan skrip *web scraping* yang telah dibuat sebelumnya [31]. Ketika halaman web berubah, skrip *scraping* mungkin perlu diperbarui atau disesuaikan agar tetap berfungsi dengan baik.

2.2.9 *Selenium*

Selenium adalah perangkat lunak *open source* yang digunakan untuk mengotomatisasi pengujian dan mengendalikan browser secara otomatis [32]. Ini adalah salah satu alat yang paling populer dan sering digunakan dalam pengendalian browser untuk tujuan seperti pengujian otomatis, *web scraping*, atau otomatisasi tugas-tugas yang berhubungan dengan browser.

2.2.9.1 Kelebihan *selenium*

Selenium mendukung beberapa bahasa pemrograman seperti *Java*, *Python*, C#, *Ruby*, dan lainnya [33]. Ini memberikan fleksibilitas kepada pengembang untuk menggunakan bahasa pemrograman yang paling nyaman bagi mereka. Selain itu, dukungan bahasa pemrograman yang luas juga berarti ada banyak sumber daya, dokumentasi, dan komunitas yang tersedia untuk membantu dalam penggunaan dan pengembangan *Selenium*.

Selenium WebDriver mendukung berbagai peramban populer seperti Google Chrome, Mozilla Firefox, Microsoft Edge, Safari, dan lainnya [33]. Dengan menggunakan *Selenium*, pengembang dapat menguji atau mengendalikan browser apa pun yang pengembang butuhkan tanpa harus mempelajari alat yang berbeda untuk setiap peramban.

2.2.9.2 Kekurangan *selenium*

Menggunakan *Selenium* membutuhkan pengaturan dan konfigurasi yang tepat [34]. Pengembang perlu mengunduh dan mengatur WebDriver yang sesuai dengan peramban yang ingin pengembang gunakan, mengelola dependensi dan pengaturan proyek, dan memastikan lingkungan yang benar untuk menjalankan skrip *Selenium*. Proses konfigurasi ini bisa rumit dan

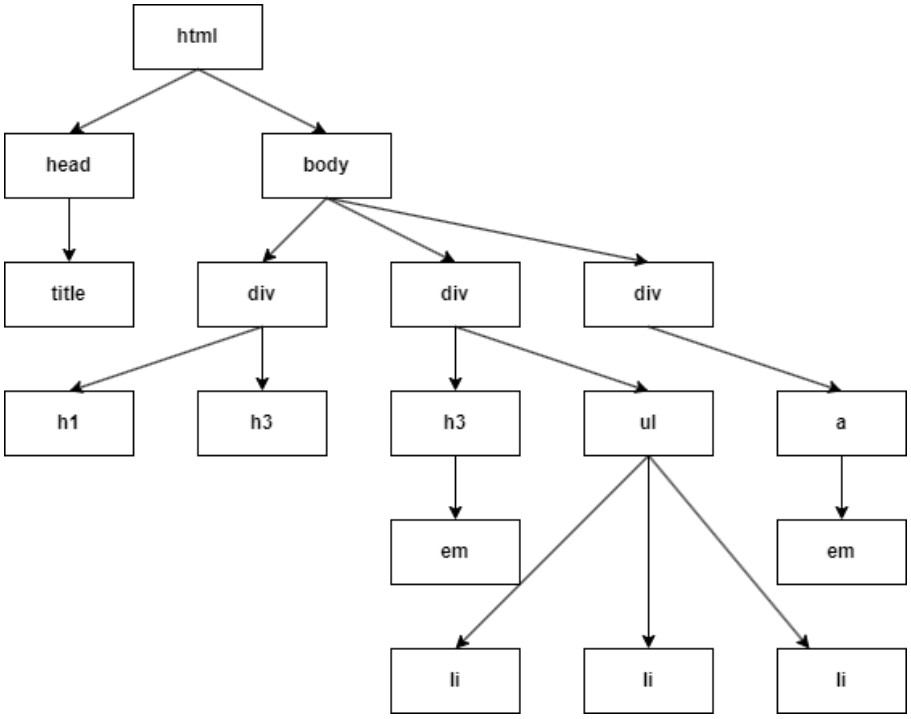
memakan waktu, terutama jika pengembang tidak memiliki pengalaman sebelumnya dengan alat ini.

Setiap WebDriver dalam *Selenium* bergantung pada peramban tertentu yang sesuai. Jika peramban mengalami pembaruan atau perubahan yang signifikan, pengembang mungkin perlu memperbarui versi WebDriver atau mengatasi masalah yang timbul akibat perubahan tersebut [34]. Hal ini dapat menyebabkan pemeliharaan dan pembaruan yang berulang pada skrip *Selenium*.

2.2.10 *Parsing HTML*

Parsing HTML adalah proses memecah struktur dari dokumen HTML untuk memahami dan mengakses elemen-elemen di dalamnya [28]. *Hypertext Markup Language* (HTML) adalah bahasa yang digunakan untuk membangun halaman web. *Parsing HTML* memungkinkan pengembang untuk mengambil data dari halaman web, mengubah tampilan halaman secara dinamis, atau melakukan tugas-tugas lain yang terkait dengan manipulasi elemen HTML. Ada beberapa cara untuk melakukan parsing HTML, salah satu cara yang umum adalah menggunakan pustaka atau *library parsing HTML* yang disediakan oleh bahasa pemrograman tertentu, seperti *BeautifulSoup* untuk *Python* atau *jsoup* untuk *Java*. Pustaka-pustaka ini menyediakan metode dan fungsi yang memudahkan pengembang dalam memparsing dokumen HTML, menavigasi struktur HTML, dan mengakses elemen-elemen di dalamnya. Dalam penelitian ini, *parsing HTML* digunakan untuk mengekstrak data tertentu, seperti nama produk, harga produk, atau informasi produk lainnya. Data ini kemudian dapat digunakan untuk analisis lebih lanjut, integrasi dengan sistem lain, atau tujuan lainnya.

Dalam melakukan *parsing HTML*, pengembang harus mengetahui dimana letak data yang ingin di *parsing,* untuk mengetahui di tag mana data berada pengembang harus melakukan *inspect element* yang terdapat pada peramban, posisi struktur tag HTML dapat dilihat pada Gambar 2.4.



Gambar 2.4 Representasi pohon dokumen HTML

2.2.10.1 Kelebihan *parsing html*

Parsing HTML memungkinkan pengembang mengekstrak data yang diinginkan dengan tepat dari elemen-elemen HTML [35]. Pengembang dapat menggunakan metode dan teknik yang sesuai untuk menemukan elemen, atribut, atau konten tertentu dalam dokumen HTML, sehingga memungkinkan pengembang untuk mengambil data dengan presisi.

Parsing HTML dapat diintegrasikan dengan berbagai pustaka dan *framework* yang mendukung parsing HTML [35]. Ada banyak pustaka dan alat yang tersedia dalam berbagai bahasa pemrograman untuk memudahkan *parsing HTML*, seperti *BeautifulSoup* untuk *Python*, *jsoup* untuk *Java*, atau *lxml* untuk bahasa pemrograman lainnya.

2.2.10.2 Kekurangan *parsing html*

Parsing HTML sangat tergantung pada struktur dokumen HTML yang sedang diurai. Jika struktur HTML berubah secara signifikan, skrip parsing mungkin perlu diperbarui atau disesuaikan agar tetap berfungsi dengan benar [35]. Hal ini dapat menjadi tantangan jika pengembang ingin melakukan

parsing pada situs web yang sering mengubah tata letak atau struktur halaman mereka.

Ketika melakukan *parsing HTML*, skrip parsing pengembang rentan terhadap kesalahan jika terjadi perubahan pada struktur atau konten halaman web yang diurai [35]. Misalnya, jika elemen yang ingin pengembang parsing belum terload datanya, skrip parsing dapat menghasilkan kesalahan atau menghasilkan data yang tidak valid.

2.2.11 *BeautifulSoup*

BeautifulSoup adalah sebuah pustaka atau *library* dari bahasa pemrograman *python* yang digunakan untuk memparsing dokumen HTML dan XML [28]. Dengan menggunakan *BeautifulSoup*, pengembang dapat mengimpor dokumen HTML dan XML ke dalam program *python* kemudian melakukan operasi seperti mengekstrak data dari elemen HTML, melakukan pencarian berdasarkan atribut atau isi elemen, menelusuri hierarki elemen, dll.

2.2.11.1 Kelebihan *beautifulsoup*

Salah satu kelebihan utama *BeautifulSoup* adalah kemudahannya dalam digunakan [11]. *BeautifulSoup* menyediakan antarmuka yang intuitif dan mudah dipahami, yang memungkinkan pengembang dengan cepat memulai *parsing* dan manipulasi dokumen HTML.

BeautifulSoup dapat memproses dokumen HTML secara menyeluruh, termasuk tag, atribut, konten, komentar, dan banyak elemen HTML lainnya [36]. Hal ini memungkinkan pengembang untuk mengekstrak data dengan presisi dan memanipulasi elemen-elemen HTML dengan mudah.

2.2.11.2 Kekurangan *beautifulsoup*

BeautifulSoup mungkin tidak secepat beberapa pustaka *parsing HTML* lainnya [11], terutama saat menghadapi dokumen HTML yang sangat besar atau kompleks. Ini bisa menjadi masalah jika pengembang perlu melakukan parsing pada halaman web yang memiliki ukuran atau struktur yang rumit.

BeautifulSoup hanya fokus pada *parsing* dan manipulasi dokumen HTML. Ini berarti bahwa *BeautifulSoup* tidak secara langsung mendukung pemrosesan *JavaScript* atau eksekusi kode *JavaScript* [37]. Jika halaman web yang ingin pengembang coba *parsing* mengandalkan pemrosesan *JavaScript* untuk menghasilkan konten, maka pengembang mungkin perlu menggabungkan *BeautifulSoup* dengan alat atau pustaka lain seperti *Selenium* untuk mengambil konten yang dihasilkan oleh *JavaScript*.

2.2.12 *Black box testing*

Black box testing adalah metode pengujian perangkat lunak yang mengarah pada fungsionalitas dari aplikasi yang bertentangan dengan struktur internal atau kerja [38]. Pada *black box testing*, pengujian dilakukan berdasarkan spesifikasi fungsional sistem, yaitu input yang diberikan dan output yang diharapkan. Tujuannya adalah untuk memvalidasi apakah sistem menghasilkan output yang sesuai dengan spesifikasi yang ditentukan, tanpa memerhatikan bagaimana sistem mencapai output tersebut.

Pengujian *Black Box* adalah suatu metode yang digunakan untuk menguji perangkat lunak tanpa memperhatikan rincian internal perangkat lunak. Dalam proses *Black Box Testing*, program yang telah dibuat diuji dengan cara mencoba memasukkan data pada setiap formulirnya. Pengujian ini dilakukan untuk memastikan bahwa program beroperasi sesuai dengan kebutuhan yang ditetapkan oleh perusahaan [39].