

# AI-Powered Web Scraping and Parsing: A Browser Extension Using LLMs for Adaptive Data Extraction

Shubham Nevgi

*Information Technology*

Finolex Academy of Management and Technology

Ratnagiri, India

[nevgishubham03@gmail.com](mailto:nevgishubham03@gmail.com)

Sahil Kadam

*Information Technology*

Finolex Academy of Management and Technology

Ratnagiri, India

[sahilk1503@gmail.com](mailto:sahilk1503@gmail.com)

Sahil Haldankar

*Information Technology*

Finolex Academy of Management and Technology

Ratnagiri, India

[sahilph46@gmail.com](mailto:sahilph46@gmail.com)

Sakshi Jadhav

*Information Technology*

Finolex Academy of Management and Technology

Ratnagiri, India

[sakshijadhav0316@gmail.com](mailto:sakshijadhav0316@gmail.com)

Prof. Rashmi More

*Information Technology*

Finolex Academy of Management and

Technology

Ratnagiri, India

[rashmi.more@famt.ac.in](mailto:rashmi.more@famt.ac.in)

**Abstract**— In the era of information overload, the need for extracting meaningful and structured data from unstructured web sources has grown significantly. Traditional web scraping tools often require significant manual effort to parse and format data, especially when dealing with complex or dynamic websites. To address this challenge, this project presents a Generative AI-based Web Scraping Browser Extension, an innovative tool that combines the power of browser automation, HTML parsing, and generative artificial intelligence to extract and interpret data intelligently. This browser extension allows users to input any URL and extract structured information from the web page using an intuitive interface. Unlike traditional scrapers that rely heavily on predefined rules or regular expressions, the system uses Generative AI models to understand the structure and context of web content. The backend, developed using FastAPI, integrates BeautifulSoup and Selenium for handling both static and dynamic web pages, while AI parsing is powered by transformer models (e.g., LLaMA 3.3). The data extracted can be visualized in a tabular format and downloaded in multiple formats, including CSV,

JSON, XML, and Excel. One of the major highlights of the system is its ability to learn patterns from previously scraped data and intelligently adapt to new page layouts, significantly reducing the need for manual intervention. This enhances productivity and provides an accessible solution for both technical and non-technical users who need structured data for research, analytics, or business intelligence.

**Keywords**— *Web Scraping, Generative AI, Data Extraction, FastAPI, Selenium, AI Parsing, Browser Extension, Automation.*

## INTRODUCTION

### A. Background:

Web scraping is vital for data harvesting in areas such as e-commerce, research, and media analysis. However, with advances in web technology, sites load content dynamically more frequently using javascript. As a result, traditional scraping techniques are unable to scrape the needed information. Most web scrapers are centralized on a fixed HTML structure which means they will need to be

manually adjusted when a website's layout changes. Additionally, numerous web scraping tools for high-resource languages mainly overlook the low-resource languages where data collection is still a hurdle. Given this increased need for more scalable and adaptable scraping tools, this project aims to enhance an existing browser extension by leveraging Selenium to handle extracting dynamic content and AI-powered parsing to sweep through and methodically extract data.

#### B. *Problem Definition:*

The main difficulty with traditional web scrapers is their limited ability to handle dynamic websites effectively, especially for low-resource languages. Most scrapers operate based on HTML structure that has been determined in advance, which makes them relatively stiff and prone to failures when websites change the structure of content. This project will address these challenges by creating a browser extension that will: Use Selenium to load content dynamically; Use Generative AI to strategically parse and extract data from websites in multiple formats; and Adjust to frequent and dynamic website changes without having to manually adjust the code since it is driven mostly by generative AI. This will enable those seeking to scrape websites to have a more scalable, flexible, and AI-driven process when scraping data from dynamic and low-resource websites that are the most difficult for traditional scrapers.

#### METHODOLOGY

We utilized Python for the backend of the Web Scraping Browser Extension, while the frontend utilized JavaScript, HTML, and CSS. The extension involved a combination of technologies that interact in a seamless way. The core scraping engine relied on adding Selenium support to scrape dynamically generated JavaScript based content and beautiful soup for parsing the static HTML elements. FastAPI was added for backend communication, while requests would handle the http requests. The traditional ways of scraping would rely on identifying patterns of fixed data structures. Our web scraping browser extension incorporates an AI-powered parser that aims to provide content extraction that is more accurate and adaptable. The web scraping browser extension was provided with a graphical interface that allows the user to input a URL, decide the data type they

want to scrape, select to scrape, view the data that is going to be scraped and preview/download scraped data in multiple formats (CSV, JSON, XML, Excel). AI and Natural Language Processing (NLP) allow the system to improve the parsing based on recognizing data and adapting to changing web structures. The development process included setting up the development environment, fine-tuning the AI model for our needs using transformers instead of tensorflow's AutoML offerings, and integrating the API/backend with the browser, thereby creating an efficient web scraping experience. The project underwent unit and integration testing as well as User Acceptance Testing (UAT) with users in the finish. Post-development work focused on making patches for multiple operating systems, documenting our work for the purpose of additional development and final development project deployment. This project ended with a final demonstration of the extension, submission of documentation, and peer review launch.

#### EXISTING SYSTEM

Current web scraping tools and extensions primarily use unknown-but-detective methods, which require the user to predefine selectors and settings to extract information from web pages. It is challenging for these mechanisms to cope with dynamic content, changing layouts and anti-scraping techniques used on websites.

#### LITERATURE SURVEY

According to the work entitled Leveraging Large Language Models for Web Scraping, by the date 2024, ArXiv in the section of LLM-based web scraping, the investigators expose the strength of LLMs (Language Models) in the aspect of data extraction and, besides that, the issues of hallucinations and reliability are indicated. In a similar vein, the research article by Generative AI Tools in Academic Research, which was published on the year 2024, ArXiv, elaborates on the transparency and trustworthiness of AI tools as the main issues and, therefore, recommends data structuring by AI over content generation. Likewise, Generative AI for Research Data Processing (2023, IEEE) sheds light on the bias that Generative AI models have and the need for a generous amount of fine-tuning to ensure precision. The study on Web Scraping Approaches and Their Performance on

Modern Websites (2022, IEEE) also mentions the limitations of conventional web scraping methods where dynamically loaded content through JavaScript is one of the problems for rule-based techniques. Previous research into Web Scraping Using Machine Learning (2020, DIVA-Portal) calls attention to the current issue of machine learning models that are not yet able to work well with different site structures as a consequence of being only able to learn from the data that is labeled. Considering the research issue, the proposed work is directed towards the design of an adaptive Generative AI-based web scraping system that is able to guarantee data fidelity, trim down manpower required for data extraction, and manage dynamic web content in the absence of extensive labeled training data.

Ref. No. & Year of Publication	References	Research Gaps	Project Research Objective
1. 2024	Leveraging Large Language Models for Web Scraping (ArXiv)	LLMs exhibit potential for data extraction but face challenges like hallucinations and reliability issues.	Use LLMs for structured information extraction to ensure data fidelity while enabling flexible user requests.
2. 2024	Generative AI Tools in Academic Research (ArXiv)	AI tools lack transparency in decision-making, raising concerns about bias and reliability for academic research.	Rely on AI for parsing and structuring web data rather than content generation, ensuring trustworthiness and factual accuracy.

3. 2023	Generative AI for Research Data Processing (IEEE)	Generative AI models introduce bias and require extensive fine-tuning for accuracy.	Focus on AI-assisted web scraping that prioritizes factual integrity and reduces manual workload.
4. 2022	Web Scraping Approaches and Their Performance on Modern Websites (IEEE)	Traditional web scraping techniques struggle with dynamic websites that load content asynchronously using JavaScript.	Explore adaptive methods that integrate Generative AI for dynamic web scraping, overcoming limitations of rule-based approaches.
5. 2020	Web Scraping Using Machine Learning (DIVA-Portal)	ML models require labeled datasets and struggle with generalization across diverse website structures.	Employ pre-trained Generative AI models for dynamic adaptation without requiring labeled training data.

TABLE 1 : LITERATURE SURVEY

OBJECTIVES

**Objective 1: Develop a browser extension for efficient data scraping from dynamic websites.**

The main aim of this undertaking is to create a browser extension that allows users to scrape data quickly from dynamic websites. These types of websites are dynamic in nature, as they render content asynchronously using JavaScript or other technologies which generate content in a dynamic manner, and regular web scraping methods do not work with them. Regular scrapers utilize static HTML to scrape data, which really relies on the HTML structure being constant (stable). With dynamic content, they simply yield inconsistent results because the content is never present on the page when the parser is completed.

Furthermore, when dynamically generating content, the DOM is typically altered in real-time after page submit or page request. Therefore, to deal with dynamic content scraping and lack of HTML to scrape on its own, we will host an extension that utilizes Selenium, an effective web browser automation tool, which will be able to interact with the page, execute whatever script, and fetch the dynamically generated content correctly.

***Objective 2: Incorporate generative AI to enhance data extraction by eliminating the need for predefined HTML tags.***

The second goal examines how to incorporate generative AI into the browser extension so that it can maximize its flexibility and efficiency in extracting data. Traditional web scraping is dependent on static fixed HTML tags and rules to find and extract data from web pages. When a website is modified in structure, the web scraper often fails, especially if the tags have changed, requiring the manual modification of the scraping rules to maintain accuracy in data extraction. This project uses a generative AI model to learn about web pages automatically, infer patterns useful for data representation, and extract useful data without the need for rule-based predefined tags. This increases the ability of the extension to adapt to changes to web pages, and reliably and accurately extract data while the web structures are changing over time.

***Objective 3: Ensure support for low-resource languages and improve accessibility to multilingual web content.***

The third objective responds to the barrier of information extraction from web pages in low-resource languages. Most automated web scraping tools are optimized to work for high-resource languages such as English, and therefore, do not perform as efficiently for data from low-resource languages. To address this problem, the proposed browser extension will be contextually designed to aid multilingual data extraction, with a particular emphasis on low-resource languages. The system will facilitate this through Natural Language Processing (NLP) techniques and Large Language Models (LLMs) capable of identifying, extracting, and processing text in a range of languages, which often includes underrepresented and poorly documented ones. Ultimately the project seeks to reduce the barrier to multilingual web engagement, thereby creating a more inclusive experience for information scraping.

## MOTIVATION

### **Perception of the Problem in the Real World and Its Benefits to Society:**

Today, the internet is a large source of useful data in the digital age. However, a vast majority of this data is unstructured, making it challenging to retrieve (and use) this information for practical purposes. Organizations, researchers and businesses utilize web data for market trend analysis, educational research, price tracking and sentiment analysis. As a result, web scraping is utilized to collect and access data that are beneficial for educators, researchers and business professionals alike, saving time and effort, manual labor and fostering data-driven decision-making. Business professionals use web scraping to understand effective marketing strategies, compare competitors' pricing, identify products with high demand, etc. While using traditional web scraping methodologies can often be hard, it can put stress on the server, causing the web application to lag in performance or use of SEO practices. For this reason, introducing an AI web scraping solution in this project will achieve a self-analyzing and self-extracting AI web-scraping methodology, eliminating the need for constant human intervention and improving efficiency, adaptability and scalability of the web scraping process. As the demand for structured web data grows, the field of AI-powered web scraping offers a paradigm shift. While existing tools depend on hard-coded rules and manual updates from users, today's state-of-the-art NLP and generative AI models can intelligently adapt to changing website architectures. This project will combine AI and automation to ease data extraction, democratize access to web data, and support a variety of applications across industries.

## PROBLEM DOMAIN

This project tackles a problem that sits at the intersection of Web Scraping, Artificial Intelligence, Data Processing and Automation. Together these domains capture the problem and scope as well as the impact of the proposed Generative AI-based Web Scraping Browser Extension. Web scraping can be used in areas such as Data Mining and Information Extraction where information is extracted from unstructured web pages into structured data. Conventional scrapers rely on rule-based methods like xpath and regex and therefore break easily when a

website updates its structure. The progress of AI (artificial intelligence) has made it easy for Large Language Models (LLMs) to understand natural language instructions and the like. In this project, the Groq's Llama 3 model, the real model, provides users with the capability of the programmable interface of taking in requirements, parsing, and scheduling. When considering Data Science and Data Engineering, the retrieved web data is generally not structured, which complicates data analysis. The project is designing by structurizing organized data in a variety of formats such as CSV, JSON, XML, and Excel, so that the data can be effortlessly analyzed and integrated with other tools. Whereas, in Software Engineering and Cloud Computing, web scraping scripts are updated manually which in turn causes them to be inaccurate. The AI-based model would be used for web scraping, which would automate the procedure, which would lead to reducing maintenance expenses and increasing scalability. This initiative focuses on the upcoming challenges in the intersection of Web Scraping, AI technologies, Data Processing, and Automation. These domains reflect the landscape and implications of the Generative AI and the generative AI-based Web Scraping Browser Extension project. Web scraping is commonly used in a variety of disciplines, including Data Mining and Information Extraction, as an approach to converting unstructured data from the web into structured data. Traditional web scrapers customarily used rules-based approaches, such as XPath and regex, which made them very fragile because they break every time a website updates its structure. Now, with the help of AI and LLMs, web scraping can be much more adaptable and intelligent. In this project, we are using Groq's Llama 3 model to enhance the parsing, scheduling, and automation of data. Importantly, AI-based web scraping can not only understand natural language-based instructions, but also adjust dynamically to changes in structures on websites, which is not measurable with traditional systems. From a Data Science & Data Engineering perspective, web-scraped data is often unstructured, meaning data analysis is difficult. Within Software Engineering and Cloud Computing, it's always broken to pay someone to maintain web scraping scripts manually, which can be time consuming, error inducing, and expensive. This project reduces maintenance needs, improves scalability, and enhances accuracy by branding the value of AI-based

automation, thus streamlining the web scraping process and introducing more flexibility.

#### PROBLEM DEFINITION

As websites are primarily HTML documents, any data they serve is likely to be reactively generated from HTML templates; therefore, traditional web scraping methods are in many ways based on arbitrary rules, XPath selectors, and regex patterns providing a brittle approach that could break if the HTML layout or structure were to change. Web scraping, in many cases, is very challenging, especially when the website loads content dynamically as it uses JavaScript to deliver content asynchronously, so often browsers like Chrome or Firefox that enable tooling for automation must be used also through libraries like Selenium, which often involve heavy overhead to commence scraping. Every one of these things, including the nature of the web scrapers, create a high barrier to entry for non-technical users, and provide a heavy workload to developers for maintenance. This web-based project proposes an extension with analysis and data-recovery tooling, powered by AI Natural Language Processing (NLP) modeling to extract data with ease and without the burden of preysing data points. It will analyze user instructions, set up FastAPI back-end with Selenium configuration through BeautifulSoup to both understand queries through graphics, and build custom language models such as Llama 3, through possible evolutions of AI ML natural language application. Ultimately, the aim is to auto-generate structured data formats, including CSV, JSON, Excel and XML data that is all entirely free from needing manual intervention, or completely opaque processes needing to be re-evaluated as they change over time.

REPRESENTATION

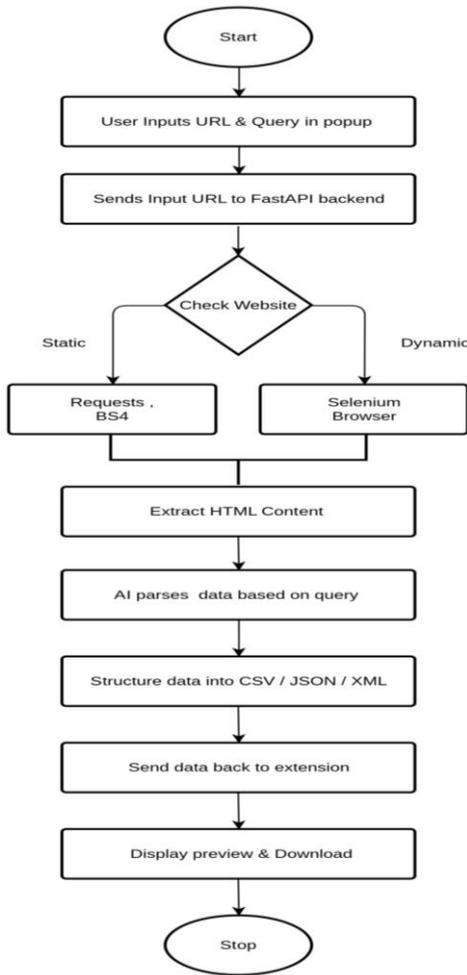


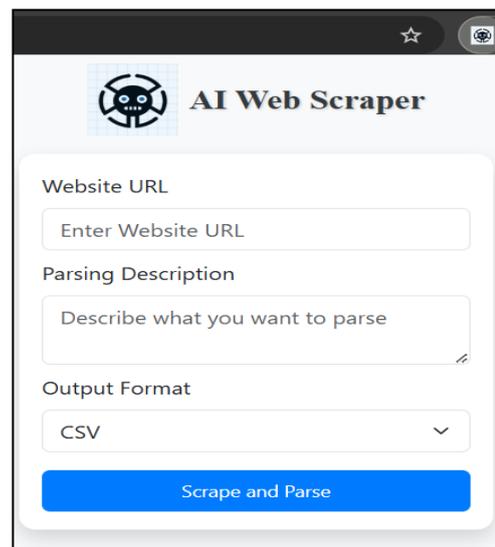
Fig 1 : Flowchart

INNOVATIVE CONTENT

1. Innovation Over Traditional Web Scraping Techniques (Reference: "Web Scraping Approaches and Their Performance on Modern Websites")
2. Innovation Over LLM-Based Data Extraction (Reference: "Leveraging Large Language Models for Web Scraping")
3. Innovation Over Machine Learning-Based Scrapers (Reference: "Web Scraping Using Machine Learning")
4. Innovation Over AI-Based Data Processing (Reference: "Generative AI for Research Data processing")

PROBLEM-SOLVING

1. Programming Methods: The system, which uses a modular programming approach, integrates multiple components to ensure efficiency, scalability, and adaptability.
2. AI-Assisted Problem Solving in Data Parsing: The main innovation of the system applied is the use of Large Language Models for AI-assisted parsing.
3. Handling Dynamic Web Content: The main problem to be solved in web scraping is the extraction of data from the JavaScript-rendered pages which are not possible to parse via the standard HTTP requests. The proposed system solves this problem by detecting JavaScript, and then switching to Selenium for browser automation, and using headless browsing mode to simulate user interactions without launching a visible browser window, which improves the performance.
4. Data Structuring and Exporting: As soon as data is obtained, it should be restructured to make it usable. This strategy ensures that the data is received by the users in a structured and usable form, thus, they are able to smoothly integrate the data with the external analysis tools.



RESULTS

Fig 2 : Ui of Extension

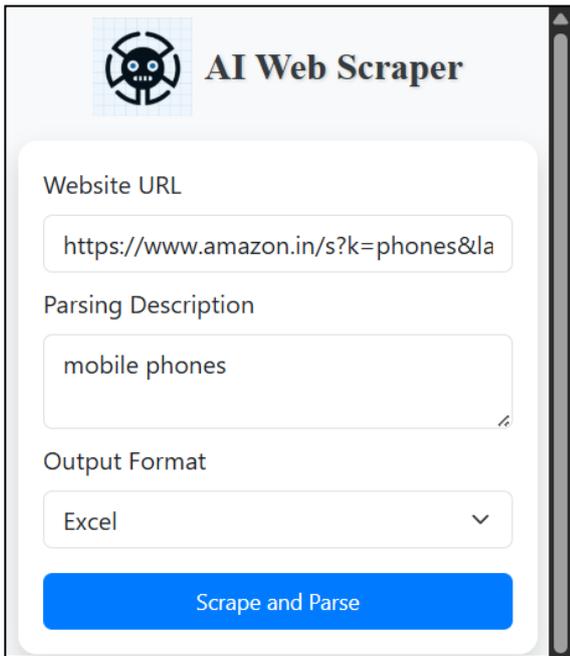


Fig 3 : Input

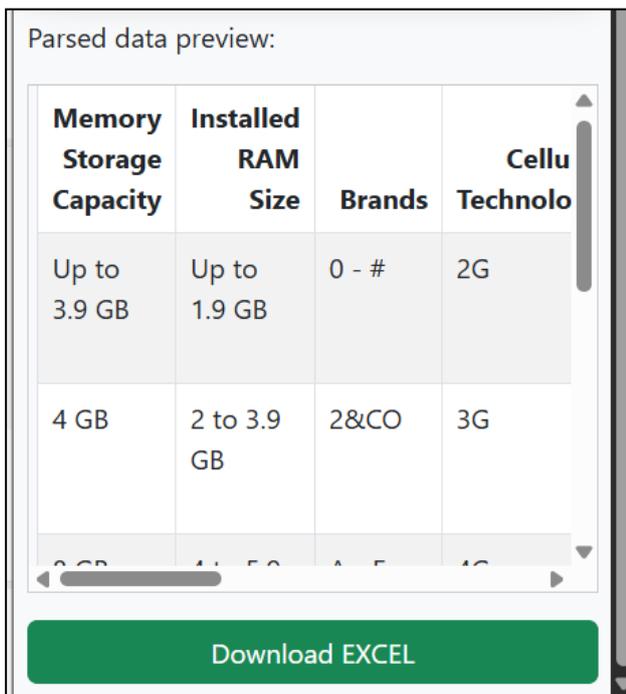


Fig 4 : Output

**CONCLUSION**

The authors introduce an AI-enabled web scraping system that utilizes Large Language Models (LLMs) to intelligently parse web content, therefore making it a more competitive option than standard web scraping techniques. Since this system is used for system data extraction, static and dynamic website data can be

automatically scraped without the need for pre-determined rules like XPath, CSS selectors, or regular expressions. Unlike normal rule-based or MLbased scrapers, the authors propose a method that adapts to structural changes in dynamic webpages and interprets user-based queries in natural language, which greatly enhances usability and scalability. The research is a bridge between AI-based Natural Language Processing (NLP) research and web scraping techniques to democratize web data extraction via scraping. The system is able to save costs associated with maintaining hardcoded extraction rules by using Generative AI to perform parsing. Future enhancements can be made to improve the system, such as improving the accuracy of the AI used to improve the quality of data extractions, decreasing latency to improve processing speed, adding multilingual support for an even wider range of users, and developing cloud-centric scraping architectures to improve large-scale data extraction scalability. In conclusion, this research has proposed a new intelligent method of web scraping, showing that LLM-assisted parsing is a more efficient, cost-effective and user-friendly approach to traditional scraping methods.

**ACKNOWLEDGMENT**

We would like to thank Finolex Academy of Management and Technology for providing the resources and support needed to conduct this research.

**REFERENCES**

- [1] "Generative AI Tools in Academic Research" (ArXiv, 2024)
- [2] "Leveraging Large Language Models for Web Scraping" (ArXiv, 2024)
- [3] Navroz Kaur Kahlon, Williamjeet Singh, (2024). "Comparative Analysis of Web Scraping Tools for Low-Resource Language Text"
- [4] "Generative AI for Research Data Processing" (IEEE, 2023)
- [5] Navdeep Singh et al.,(2023) "DeepSpacyNER: An Efficient Deep Learning Model for Named Entity Recognition for Punjabi Language".

- [6] "Web Scraping Approaches and Their Performance on Modern Websites" (IEEE, 2022)
- [7] Fantahun Gereme et al., (2021). "Combating Fake News in 'Low-Resource' Languages: Amharic Fake News Detection Accompanied by Resource Crafting"
- [8] "Web Scraping Using Machine Learning" (DIVA-Portal, 2020)
- [9] S. Zheng, D. Wu, R. Song, and J.-R. Wen, (2017). "Wrapping oriented classification of web pages".
- [10] M. Spaanem, (2017). "Static vs. dynamic websites: What are they and which is better? — rocket media".
- [11] Shaharia Islam Rabby (2017). "The Web Application Based on Web Scraping"