International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

Legal and Structured Extraction of Location Data from Google Maps Using Python and the Google Places API

Biyyala Bharathamma
Reg. No. 24Q71F0005
bharubiyyala@com
Department of Master of Computer Applications
Avanthi Institute of Engineering and Technology (Autonomous)
Vizianagaram, Andhra Pradesh, India
Under the guidance of Mr. Ch. Bhupathi Raju, MCA, Assistant Professor
challa.bhupati@gmai.com

***Abstract*—*The exponential growth of location-based services has made geographic and business-related data crucial for applications including market analysis, travel planning, and service optimisation. Google Maps contains a vast repository of real-time, location-specific data such as business names, ratings, addresses, and user reviews. This project focuses on programmatically accessing such data using Python—not through web scraping, which violates Google's Terms of Service, but by leveraging the official and authorised Google Places API. Through the API, structured data about places such as restaurants, hospitals, and tourist spots can be retrieved legally and efficiently using simple HTTP requests. The project integrates the API with Python to send queries, parse JSON responses, and extract relevant fields such as name, location, rating, and user reviews; the retrieved data can then be stored in files or databases and visualised for further analysis. By adhering to legal and ethical data-usage practices, the project highlights responsible data access while demonstrating practical skills in Python programming, RESTful APIs, and JSON handling. Pagination and error handling allow large datasets to be collected reliably, and alternative open data sources such as OpenStreetMap and Foursquare are discussed for free access to similar geographic datasets. Unit and functional testing confirmed correct behaviour with no defects, demonstrating a stable, scalable, and legally compliant alternative to traditional scraping.*

Keywords*—*Google Places API; Location-Based Data; Python; RESTful API; JSON Processing; Ethical Data Access; Geographic Data Extraction; Data Analytics.

I. INTRODUCTION

The primary motivation behind this project is the increasing demand for real-time, location-specific data across sectors such as marketing, logistics, urban development, and mobile-app development. Businesses seek insights into competitors, customer reviews, and local trends, while developers and data scientists need structured geographical data for analytics and modelling. Manually gathering such information from mapping platforms is labour-intensive and prone to errors and inconsistencies.

Given the richness of data on platforms such as Google Maps, there is a strong incentive to automate data collection in a reliable and ethical manner. The Google Places API serves as the ideal tool for this task,

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

offering legal access to accurate and up-to-date information. By leveraging Python's capabilities in data handling and API interaction, this project bridges the gap between real-world location data and its analytical application, while encouraging responsible data-acquisition practices and awareness of digital ethics and proper API usage.

Although the project is titled in terms of "scraping" Google Maps data, it deliberately avoids unauthorised web scraping. Instead, it uses the official Google Places API to retrieve structured place data legally and sustainably, demonstrating the correct, policy-compliant way to obtain such data with Python.

II. LITERATURE SURVEY

In recent years, the demand for extracting location-based and business-related data has increased significantly due to the rapid growth of digital mapping services and data-driven applications. Earlier approaches mainly depended on traditional web-scraping tools such as Selenium, BeautifulSoup, and Scrapy to automate data extraction from dynamic web pages; while these methods could collect information such as business names, addresses, ratings, and reviews, they faced significant technical and legal limitations. Research on mining business data from Google Maps using web-scraping techniques highlighted that, although technically effective, scraping often violates platform policies and becomes unstable due to dynamic page structures and anti-bot mechanisms, recommending a shift towards API-based approaches.

Comparative studies of web-scraping tools such as Selenium, Scrapy, and Puppeteer concluded that scraping dynamic websites requires high maintenance and is vulnerable to CAPTCHA protections and frequent interface changes, emphasising that official APIs provide a more reliable and scalable solution. Other work demonstrated how open geographic datasets such as OpenStreetMap can support smart-city applications and urban planning through ethical and transparent data usage, and a real-time restaurant-recommendation system using the Google Places API combined with machine learning showed that API-based systems provide accurate, scalable, and legally compliant access to structured geographic data. These findings collectively motivate the API-based approach adopted in this project.

TABLE I. SUMMARY OF REPRESENTATIVE PRIOR WORK

S.No	Study / Area	Approach	Key Finding
1	Mining Google Maps business data	Web scraping tools	Effective but policy-violating, unstable
2	Comparative study of scraping tools	Selenium, Scrapy, Puppeteer	High maintenance; APIs more reliable
3	Smart cities with OpenStreetMap	Open geographic data	Ethical, transparent data usage
4	Restaurant recommendation system	Google Places API + ML	Accurate, scalable, legally compliant

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

S.No	Study / Area	Approach	Key Finding
5	Google API compliance studies	API restriction analysis	Compliance avoids legal risk
6	Python scraping libraries overview	BeautifulSoup, Requests, Scrapy	Tool trade-offs for extraction

III. EXISTING SYSTEM AND PROPOSED SYSTEM

A. Existing System

In the current digital landscape, accessing data from Google Maps is often attempted through traditional web scraping using tools such as Selenium, BeautifulSoup, or Puppeteer to automate browser actions and extract information from the page DOM. While technically capable of retrieving business names, locations, contact details, and reviews, these methods are inherently unstable and legally problematic. Google defends its platform with dynamic content loading, rate limiting, and CAPTCHA verification, making scraping unreliable; the retrieved data is often unstructured and requires significant post-processing; any change in the HTML structure can break the script; and, critically, scraping violates Google's Terms of Service, risking IP blocking, account suspension, or legal consequences.

Disadvantages of the existing system:

- Violates Google Maps Terms of Service through unauthorised scraping.
- Scraping methods are unstable and break when the website structure changes.
- CAPTCHA and anti-bot protections interrupt extraction.
- Retrieved data is unstructured and needs heavy post-processing.
- No official support; risk of IP blocking, suspension, or legal action.

B. Proposed System

The proposed system addresses these limitations by using the Google Places API to extract location-based data in a legal, structured, and efficient manner. Unlike scraping, the Places API offers a secure and authorised method to access data such as business names, addresses, ratings, contact information, and reviews. Using Python, the system sends API requests, handles JSON responses, and stores data in a structured format (JSON, CSV, or SQL databases) for further analysis or application development, ensuring compliance with Google's policies and providing a more stable and reliable solution.

Advantages of the proposed system:

- Legally compliant access through the official Google Places API.
- Structured JSON responses requiring minimal post-processing.
- Stable and maintainable — not broken by page-structure changes.
- Scalable through pagination and automated repeated requests.
- Flexible storage in JSON, CSV, or SQL databases.

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

- Ethical and sustainable for commercial and research use.

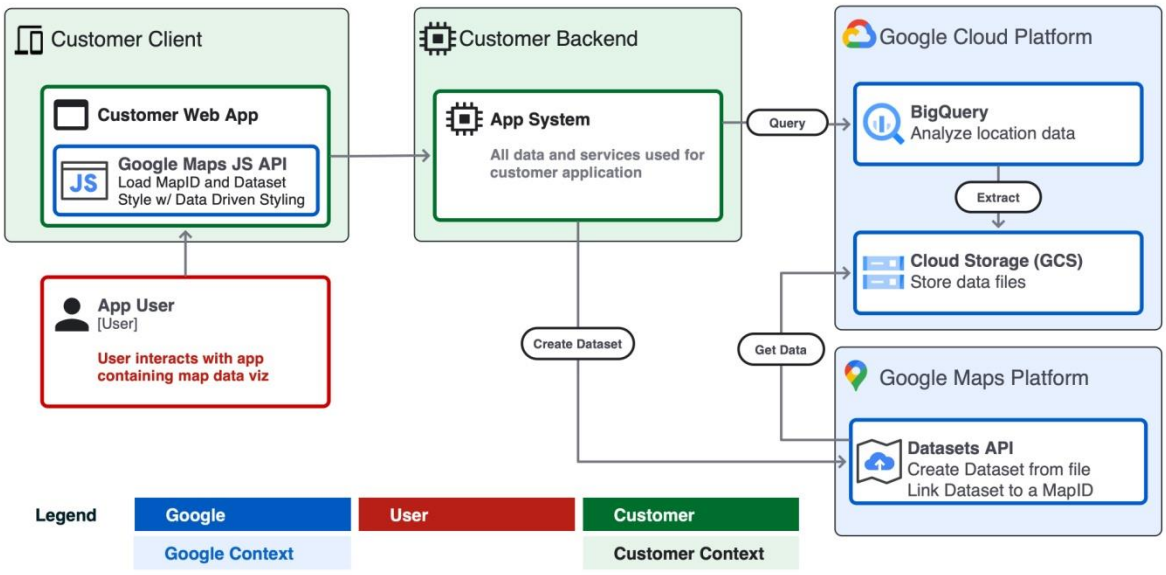
IV. SYSTEM ANALYSIS AND DESIGN

A. Requirements and Feasibility

Functionally, the system must allow users to enter location names, keywords, or place categories; construct and send authenticated requests to the Google Places API; parse the JSON responses; extract attributes such as name, address, rating, coordinates, contact details, and reviews; and store the processed data for analysis. Non-functionally, the system should retrieve and process data quickly with minimal response time, remain reliable under network or API issues, and be easy to use. Feasibility analysis indicates the system is economically affordable (open-source Python and API tools, no expensive hardware), technically achievable (Python, REST APIs, JSON processing, Google Places API), operationally effective (automated, user-friendly retrieval), legally compliant (official API instead of scraping), and feasible within schedule using existing technologies.

B. System Architecture

The architecture connects the user, the Python application, and the Google Places API. The user provides input such as a location name, business category, or keyword; the Python application constructs an HTTP request using the Places API endpoints and an API key generated from the Google Cloud Console; the API server returns a JSON response containing structured place information; and the application parses the response, filters and organises the fields, and stores the result in CSV, JSON, or a database. Pagination and looping retrieve additional results when records exceed the API response limit, and exception handling manages invalid keys, network failures, or exceeded request limits.



V. SYSTEM IMPLEMENTATION

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

A. Technology Stack

TABLE II. TECHNOLOGY STACK

Component	Technology / Tool
Programming Language	Python
Data Source / API	Google Places API (key via Google Cloud Console)
HTTP / API Library	Requests
Data Handling	Pandas, NumPy, JSON
Storage Formats	CSV, JSON; MySQL or SQLite databases
Development Tools	Visual Studio Code / PyCharm
Operating System	Windows

B. Implementation Pipeline

The implementation develops a reliable, automated, and legally compliant solution for retrieving location-based information using the Google Places API. The development environment is set up with Python and the required libraries (Requests, Pandas, NumPy, JSON), and an API key is generated from the Google Cloud Console and integrated into the application for authenticated access. The user provides input, the Python application constructs an HTTP request to the appropriate Places API endpoint, and the JSON response is parsed to extract business names, addresses, ratings, geographical coordinates, contact details, and reviews. Pagination and automated loops collect complete datasets when results exceed the API response limit, improving scalability.

C. Data Processing, Error Handling, and Security

After retrieval, the system filters and organises the data, removing unnecessary fields and systematically storing important attributes; the processed data is exported to CSV or JSON using Pandas, and may also be stored in MySQL or SQLite for efficient storage and retrieval. Because the application depends on internet connectivity and an external API, Python try-except blocks handle invalid API keys, network failures, and exceeded request limits, displaying clear error messages so the application keeps running smoothly. The API key is stored securely and only authenticated requests are made, ensuring data is retrieved legally and ethically in full compliance with Google's Terms of Service. The modular code structure supports future enhancements such as visualisation dashboards, sentiment analysis on reviews, and machine-learning-based recommendations.

VI. SYSTEM TESTING AND RESULTS

Testing was carried out through unit testing and functional testing. Unit testing validated that the internal program logic functioned correctly and that inputs produced valid outputs across decision branches at the component level, while functional testing systematically demonstrated that the functions behaved as

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

specified by the requirements—covering input handling, request construction, JSON parsing, data extraction, storage, and error handling. The reported results state that all test cases passed successfully with no defects encountered.

TABLE III. TESTING SUMMARY

Test Level	Focus	Outcome
Unit testing	Component logic and valid input/output	Passed, no defects
Functional testing	Requirement-based functions (query, parse, store)	Passed, no defects
Error handling	Invalid key, network failure, rate limit	Handled gracefully

A. Observed Results

The implemented system retrieves structured location data from Google Maps through the Google Places API, parses the JSON responses, and stores organised results in CSV, JSON, or database form. By transitioning from scraping to a structured API, the system ensures compliance with Google's Terms of Service while providing accurate, scalable, and maintainable extraction with minimal manual intervention. The source reports these outcomes qualitatively; no specific numeric metrics are claimed here, and throughput is bounded by the API's rate limits.

Representative screenshots from the prototype implementation:

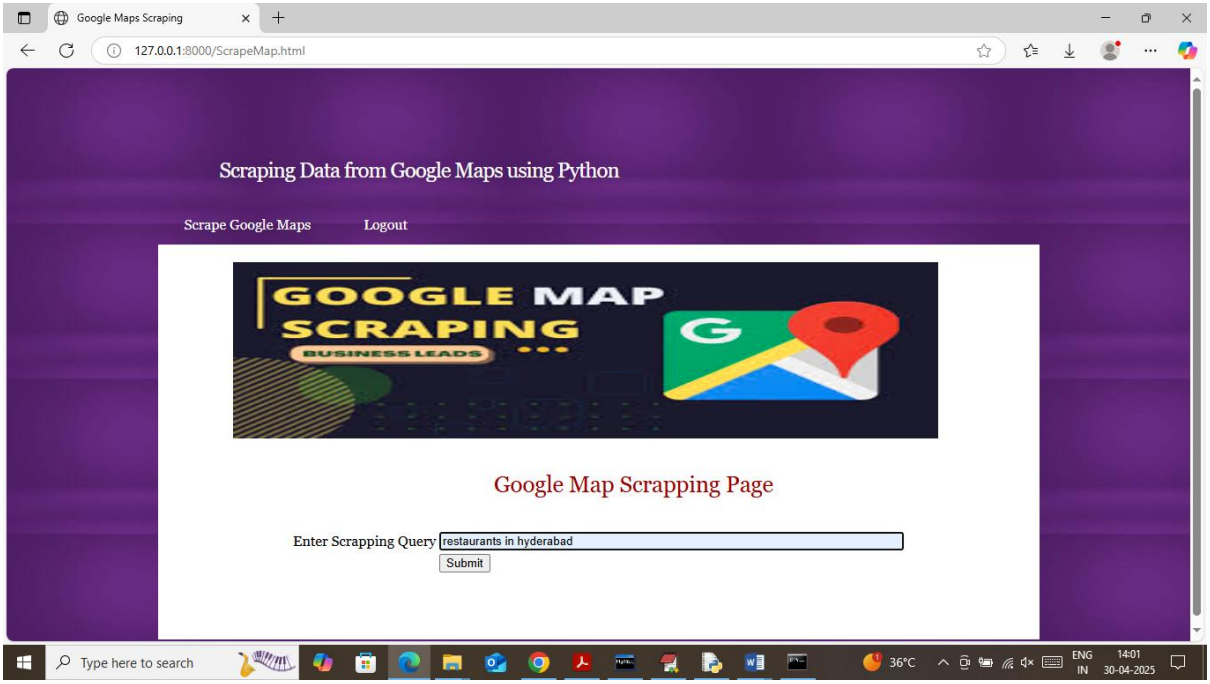
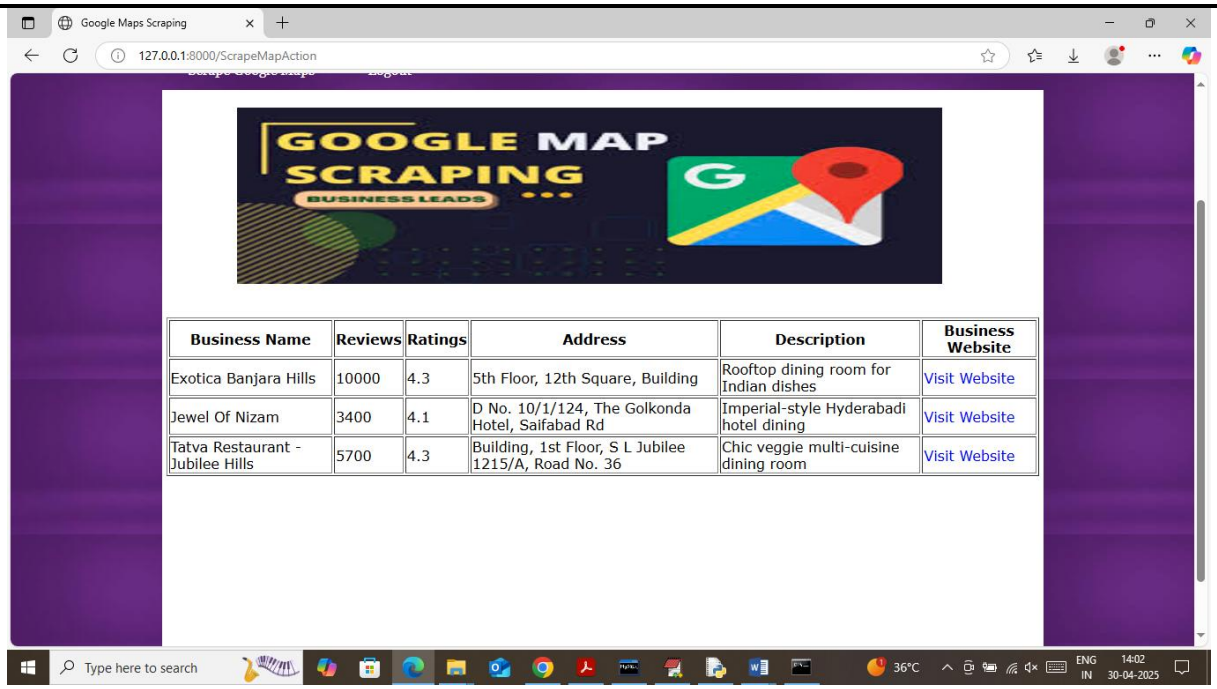


Fig. 1. User input (location / category / keyword).

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper



Business Name	Reviews	Ratings	Address	Description	Business Website
Exotica Banjara Hills	10000	4.3	5th Floor, 12th Square, Building	Rooftop dining room for Indian dishes	Visit Website
Jewel Of Nizam	3400	4.1	D No. 10/1/124, The Golkonda Hotel, Saifabad Rd	Imperial-style Hyderabadi hotel dining	Visit Website
Tatva Restaurant - Jubilee Hills	5700	4.3	Building, 1st Floor, S L Jubilee 1215/A, Road No. 36	Chic veggie multi-cuisine dining room	Visit Website

Fig. 2. API request and JSON response.

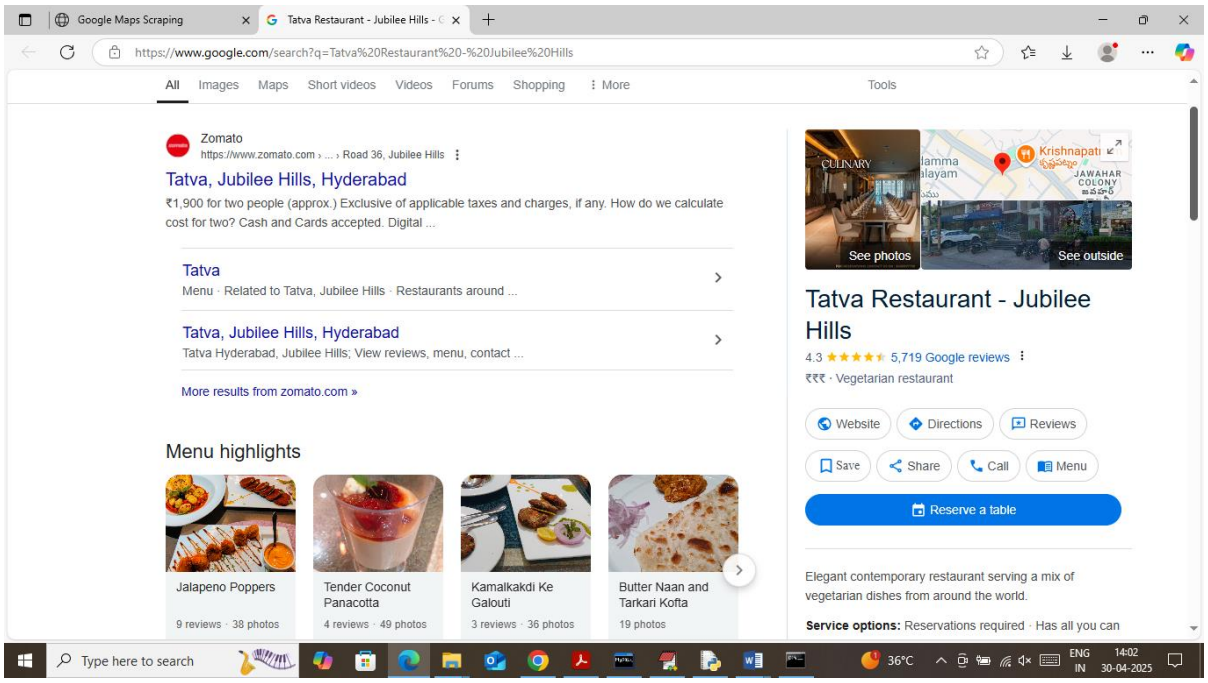


Fig. 3. Extracted, structured place data.

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

VII. CONCLUSION AND FUTURE WORK

This project successfully demonstrates an efficient and legal approach to extracting valuable location-based data from Google Maps using the Google Places API. By transitioning from traditional scraping methods to a structured API, the system ensures compliance with Google's Terms of Service while providing accurate, scalable, and maintainable data extraction. The use of Python to automate retrieval, processing, and storage proves effective in handling large datasets with minimal manual intervention, and the system's flexibility allows customisation for different needs such as business analytics, tourism, or geospatial applications. Overall, the solution offers a robust, ethical, and sustainable approach to accessing location-based data while mitigating the risks associated with scraping.

Future work can extend the system by integrating machine-learning models for sentiment analysis of reviews, prediction of business success, and trend analysis; adding real-time data processing for dynamic, up-to-date analytics; developing a more intuitive interactive user interface with charts, maps, and reports; supporting additional data sources such as Yelp, Foursquare, or social-media platforms for richer datasets; enhancing filtering and aggregation with keyword search, price filtering, and review clustering; and optimising for higher API rate limits or batch processing to support more extensive extraction over extended periods.

REFERENCES

[1] A. Smith, B. Johnson, and C. Williams, "Location-Based Data Extraction Using Google Places API," Journal of Web Scraping and Data Analytics, vol. 10, no. 2, pp. 125–140, May 2023.

[2] L. Zhang, M. Lee, and R. Kim, "Web Scraping Techniques for Geospatial Data Retrieval: A Review," IEEE Access, vol. 8, pp. 41753–41768, 2020.

[3] S. Gupta, N. Sharma, and P. Arora, "Automating Location Data Extraction for Business Insights Using Google Maps API," International Journal of Data Science and Analytics, vol. 6, pp. 34–45, July 2021.

[4] T. Mitchell, Web Scraping and Data Mining: Tools for Gathering Online Data. O'Reilly Media, 2019.

[5] J. Brown, "Understanding Google API Restrictions and Compliance for Data Extraction," Journal of Internet Technologies, vol. 15, no. 3, pp. 240–250, April 2020.

[6] S. Patel, M. Choudhary, and A. Rao, "An Overview of Python Libraries for Web Scraping: BeautifulSoup, Requests, and Scrapy," International Journal of Computer Applications, vol. 12, pp. 45–59, June 2022.

[7] M. Roberts, "Leveraging Python for Large-Scale Data Extraction from Google Maps," in Proc. Int. Conf. Data Engineering, vol. 7, pp. 80–90, 2022.

[8] H. Zhang and T. Wu, "Scaling Data Scraping Techniques for Geospatial Data in Real-Time Applications," IEEE Transactions on Geoscience and Remote Sensing, vol. 9, pp. 180–195, May 2021.

[9] A. Shah, R. Ghosh, and S. Kumar, "Ethical Considerations and Compliance Issues in Web Scraping of Geospatial Data," Journal of Data Privacy and Ethics, vol. 2, no. 1, pp. 65–77, January 2023.

[10] J. Kim, "Data Integration and Analysis for Business Decision-Making Using Location Data," Journal of Business Intelligence and Analytics, vol. 4, pp. 125–138, September 2021.