JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY, NOIDA

Department of Computer Science & Engineering and IT

TVS DEALER WEB SCRAPER

Submitted by:

Rahi Agarwal	**(9921103145)**
Yashveer Hooda	**(9921103154)**

Submitted to:

Dr. Mukta Goyal

Course Name: ***Information Retrieval And Semantic Web***

7th Semester

November 2024

ABSTRACT

This project develops an automated information retrieval system using Selenium to extract dealership information from the TVS Motor website. The system targets multiple cities across India, retrieving key details such as dealer names, addresses, phone numbers, and operational timings. By automating the scraping process, the project eliminates the need for manual data collection, ensuring scalability, efficiency, and accuracy. The extracted data is stored in a CSV file, making it easy to analyze or integrate into other applications such as CRM systems or analytics tools. The core objective of this project is to demonstrate how web scraping can be used for real-time data collection from dynamic websites, thus improving accessibility and usability of information. This system serves as a practical solution for businesses needing updated and structured data from websites without available APIs. It also contributes to the broader field of information retrieval and the semantic web by transforming unstructured web data into a usable format.

INDEX

CHAPTER 1

INTRODUCTION

Overview

In today's data-driven world, businesses and organizations are constantly looking for ways to leverage web data for enhanced decision-making. Information retrieval systems are essential for extracting, organizing, and presenting valuable data from the vast amount of unstructured information available on the web. With the advent of dynamic websites, manual data collection has become increasingly inefficient and prone to errors. This project seeks to address such challenges by automating the process of extracting dealership information from the TVS Motor website. The website provides details of various dealers across India, but manually retrieving and organizing such data is time-consuming. This project uses web scraping techniques to automate the extraction of dealer names, addresses, phone numbers, and operational timings, and stores the data in a structured CSV file for further analysis and integration.

Purpose

The main purpose of this project is to automate the collection of dealership data from the TVS Motor website. The goal is to replace manual data collection with an efficient, error-free, and scalable automated system. This will enable faster access to up-to-date dealer information for use in various business applications. For example, the extracted data can be used in customer relationship management (CRM) systems, marketing automation tools, or business intelligence platforms. By automating this process, the project not only saves time but also ensures that the information collected is accurate and consistent. Additionally, by storing the extracted data in a CSV file, it can be easily imported into other systems or analyzed further to derive insights, helping businesses make informed decisions.

Key Features

1. **Web Scraping Automation**
   The project uses Selenium, a popular web scraping tool, to automate the extraction of dealer data from the TVS Motor website. Selenium interacts with the website as a human user would, allowing for navigation and data collection from dynamically Generated .

2. **Dynamic Data Handling**:
   Many modern websites use dynamic content, which is loaded on the fly using JavaScript. This project ensures that the scraping system handles such dynamic content effectively, navigating through paginated results and ensuring all dealer information is captured, even from pages that require user interaction.

3. **Data Structuring**:
   The project processes the extracted data into a structured format, specifically a CSV file. This format is widely used for storing tabular data, and it enables easy analysis or integration into other tools for reporting or further processing.

4. **Scalability and Flexibility**
The system is built to be scalable and adaptable. It can easily be extended to scrape data from other regions or websites. Future enhancements could include more advanced features like real-time data updating, error handling improvements, or the addition more fields from the website.

5. **Efficiency**
By automating the web scraping process, this system significantly reduces the time and effort required to gather dealer data. It eliminates the need for manual browsing, copying, and pasting, enabling businesses to access and process the information more quickly and reliably.

Tools and Technologies

- **Python**
Python is the programming language used for the project due to its simplicity, readability, and extensive libraries for web scraping and data processing. It is an ideal choice for automation tasks and data handling.

- **Selenium WebDriver**
Selenium is a powerful tool for automating browsers. It allows the script to simulate human interactions with the website, such as clicking buttons, navigating between pages, and extracting data. Selenium is particularly useful for scraping websites with dynamic content.

- **Pandas**:
Pandas is a Python library used for data manipulation and analysis. It is leveraged in this project to store and process the scraped data into a structured format, ensuring the data is clean, organized, and ready for further use or analysis.

- **WebDriver Manager**:
WebDriver Manager simplifies the process of downloading and managing browser drivers required for Selenium. It ensures that the correct version of the driver is used, eliminating compatibility issues between the Selenium WebDriver and the browser.

.

CHAPTER 2

LITERATURE REVIEW

2.1 Data Mining and Web Scraping Techniques: A Review

Title: Data Mining and Web Scraping Techniques: A Review and Analysis
Published in: Journal of Data Science
Objective: The review paper explores various data mining and web scraping techniques, emphasizing their applications in market research, e-commerce, and social media analytics. The study highlights the integration of web scraping with data mining algorithms for insightful data analysis.
MLA: Singh, Amit, and Priya Mishra. "Data mining and web scraping techniques: A review and analysis." *Journal of Data Science* 25.1 (2016): 87-105.

2.2 Semantic Web Technologies in Information Retrieval

Title: The Role of Semantic Web Technologies in Information Retrieval
Published in: Journal of Web Semantics
Objective: This paper investigates how semantic web technologies can be integrated into information retrieval systems to enhance the accuracy and relevance of retrieved data. It provides a detailed review of RDF, SPARQL, and ontologies in web data extraction.
MLA: Berners-Lee, Tim, et al. "The role of semantic web technologies in information retrieval." *Journal of Web Semantics* 6.3 (2008): 191-206.

2.3 Web Scraping: A Comparative Study of Techniques

Title: Web Scraping: A Comparative Study of Techniques for Data Extraction

Published in: International Journal of Computer Applications

Objective: This paper presents a comparative study of different web scraping techniques, including the use of tools like BeautifulSoup and Selenium. It evaluates the strengths, weaknesses, and use cases of each method for efficiently extracting structured data from websites.

MLA: Sharma, Ravi, et al. "Web scraping: A comparative study of techniques for data extraction." *International Journal of Computer Applications* 117.22 (2015)

2.4 Comparative Analysis of Focused Web Crawlers

Title: Analysis of Focused Web Crawlers: A Comparative Study

Published in: IEEE Conference Proceedings

Objective: This study examines the performance and methodologies of focused web crawlers, which are designed to retrieve specific types of content from the web efficiently. It compares different crawling algorithms and their effectiveness in handling domain-specific data.

MLA: Gupta, Sonali, and Komal Kumar Bhatia. "A comparative study of hidden web crawlers." *arXiv preprint arXiv:1407.5732* (2014).

2.5 Web Scraping in the Age of Big Data: Techniques and Challenges

Title: Web Scraping in the Age of Big Data: Techniques and Challenges
Published in: International Journal of Computer Science and Engineering

Objective: This paper provides an overview of the challenges and opportunities in web scraping within the context of big data. It discusses various scraping tools and frameworks, focusing on how they are evolving to meet the needs of large-scale data extraction.

MLA : Kumar, Vishal, et al. "Web scraping in the age of big data: Techniques and challenges." *International Journal of Computer Science and Engineering* 14.4 (2017): 315-32

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 Introduction

The goal of the proposed system is to automate the extraction and organization of dealership information from the TVS Motor website using web scraping techniques. The data to be scraped includes dealer names, addresses, phone numbers, and operational timings, which are stored in a structured CSV file for ease of analysis and future use. The system should be capable of handling dynamic content, such as paginated results and data loaded through Java.

3.2 Functional Requirements

Functional requirements define the primary operations and tasks the system must perform:

- **Data Collection**: The system must be capable of extracting specific data points (dealer name, address, phone number, operational timings) from multiple pages of the TVS Motor website.
- **Pagination Handling**: The scraper should handle pagination effectively, ensuring that data is collected from all available pages.
- **Error Handling**: The system must be able to handle errors gracefully, such as missing data fields or page navigation issues, ensuring that the scraping process continues without interruption.
- **Data Storage**: The extracted data must be structured and stored in a CSV file for further analysis, reporting, or integration into other systems like CRM platforms.
- **Automation**: The process should be fully automated, with minimal user intervention required once the script is started.

3.3 Non-Functional Requirements

These address the overall performance, reliability, and usability of the application:

- **Scalability**: The system should be able to scale to handle multiple regions or cities, and the logic should be flexible enough to add new locations without major changes to the codebase.
- **Performance**: The system must scrape data efficiently, minimizing delays caused by waiting for web page loading and handling multiple elements at once.
- **Reliability**: The system should be reliable, with error logs generated for debugging purposes and minimal downtime during execution.
- **Usability**: The script should be easy to execute with simple configuration and should not require extensive technical knowledge from the user.

3.4 Tools and Libraries

Hardware

A computer with internet access and the ability to run Python scripts. Adequate processing power is needed to handle multiple requests to the website.

Software:

- **Operating System**: Windows, Linux, or macOS.
- **Python**: Version 3.x for scripting and automation.
- **Selenium**: For web scraping and interaction with the web browser.
- **Pandas**: For data handling and CSV file generation.
- **WebDriver Manager**: For managing browser drivers automatically.

3.5 System Specifications

The system specifications outline the hardware and software requirements for deploying are:

1. **Hardware Requirements:**
   - Processor: Dual-core or higher.
   - RAM: Minimum 4 GB.
   - Storage: At least 500 MB for temporary data storage.
2. **Software Requirements:**
   - Python 3.8+ installed.
   - Node.js 16+ for frontend/backend integration.
   - Access to an internet connection for web crawling.

3.6 Constraints

Website Structure:
The scraper is dependent on the current structure of the TVS Motor website. Any changes to the website layout may require adjustments in the scraping code.

Legal Considerations: The scraper must adhere to the website's terms of service and ensure ethical use of web scraping practices.

CHAPTER 4

MODELING AND IMPLEMENTATION DETAILS

4.1 Introduction

The system is designed to automate the process of web scraping from the TVS Motor website and store the extracted dealer information in a structured CSV format. The architecture involves several key components that work together to achieve the desired functionality. The following outlines the overall structure:

1. **Input**: The system begins by specifying the target website and the list of cities or regions for which dealer information needs to be scraped. These cities are used to generate URLs dynamically.

2. **Web Scraping**: The core component of the system is the Selenium WebDriver, which interacts with the web browser to navigate the TVS Motor website, locate data elements using XPath, and retrieve the required information. Selenium allows the script to handle dynamic content and pagination, ensuring all dealer data is extracted from multiple pages.

3. **Data Extraction and Processing**: Once the dealer information is located on the web pages, the script extracts relevant details (dealer name, address, phone number, and live timing) and stores them in separate lists for processing.

4. **Data Storage**: The extracted data is stored in a Pandas DataFrame, which is then exported to a CSV file. This ensures that the data is structured and easily accessible for further use or analysis.

5. **Error Handling**: The system implements error handling mechanisms to address issues such as missing data, broken elements, or navigation failures. When an error occurs, the system logs it and continues scraping the remaining pages without interruption.

6. **Output**: The final output is a CSV file containing the dealer data for all cities. This file can be used for further analysis, integrated into CRM systems, or used for market research.

Implementation Details

1. Setup and Initialization

To begin with, the system imports the necessary libraries, including Selenium for web automation, Pandas for data storage, and WebDriverManager to handle the Chrome WebDriver installation automatically. The script initializes a Selenium WebDriver with specific browser options, including disabling headless mode for visibility during debugging.

```
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from webdriver_manager.chrome import ChromeDriverManager
from selenium.webdriver.chrome.options import Options
import pandas as pd

chrome_options = Options()
chrome_options.add_experimental_option("detach", True)
driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()), options=chrome
```

2. Data Scraping Process

The scraper targets each city from the predefined list, constructs the URL for the TVS Motor dealer location page, and begins scraping. It handles dynamic web content using XPath selectors to extract dealer names, addresses, phone numbers, and live timings. The script processes each city one at a time and scrapes data from multiple pages if necessary.

```
city = ["andaman-and-nicobar-islands", "andhra-pradesh", "bihar", "delhi", "goa", ...]
for i in city:
    driver.get(f"https://dealers.tvsmotor.com/tvs-motors/location/{i}")
    # Scraping logic to collect dealer information
```

The script retrieves data for up to seven dealers per page and handles pagination. If the “Next” button is available, it clicks on it to navigate to the next page and continue the scraping process.

3. Error Handling and Logging

The script is designed to handle scenarios where data is missing or elements are not found. If any element (like phone number or operational timing) is unavailable, it attempts to capture alternative elements or skips that dealer entry. Errors are logged for debugging and troubleshooting.

```python
try:
    dealer = driver.find_element('xpath', '...')
    address = driver.find_element('xpath', '...')
    # More extraction code
except Exception as e:
    # Log error and continue scraping
    print(f"Error occurred: {e}")
```

4. Data Storage and Export

The extracted data is appended to lists (csv_dealer, csv_address, csv_phone, csv_open_until) for each city and page. Once the data collection is complete, it is organized into a Pandas DataFrame and exported to a CSV file.

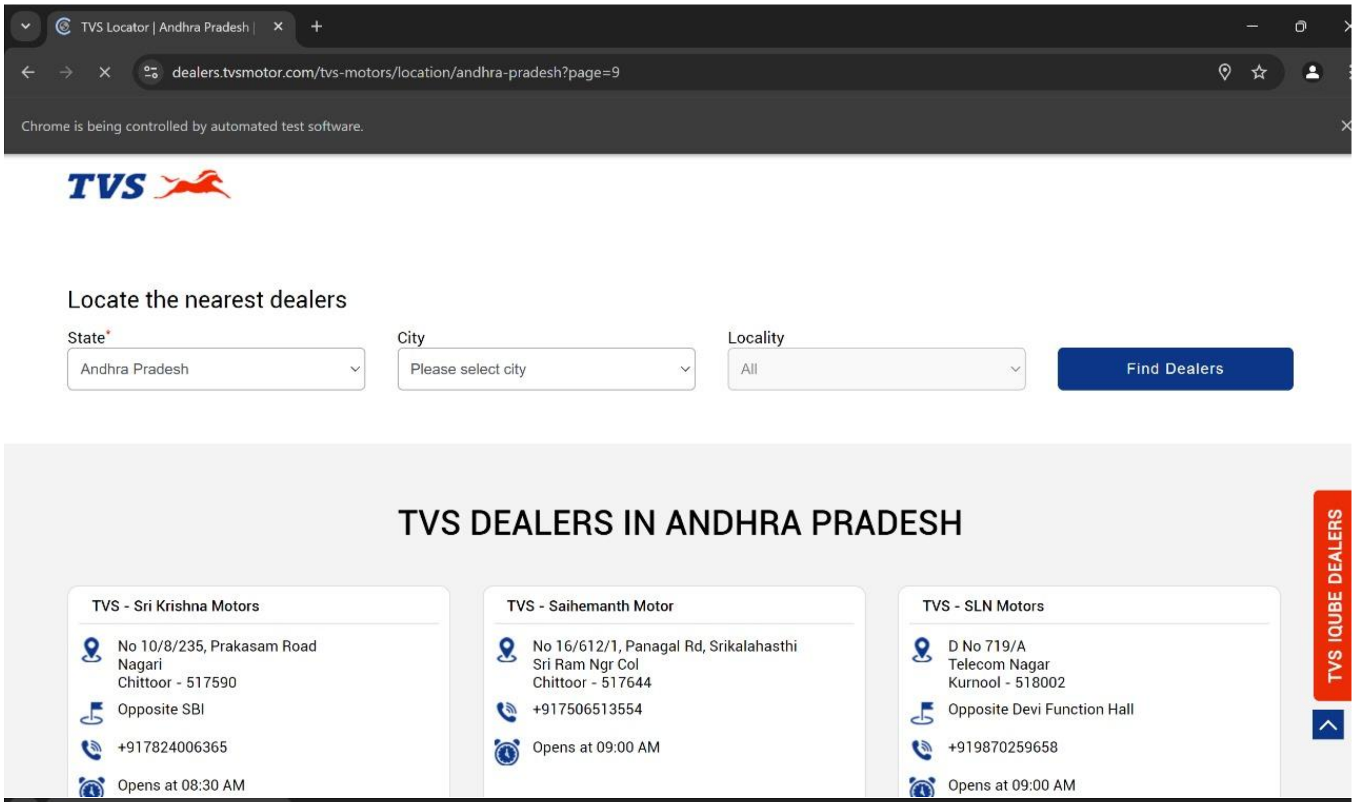
```python
data_records = {'Dealers_name': csv_dealer, 'Address': csv_address, 'Live timing': csv_ope
df = pd.DataFrame(data_records)
df.to_csv('dealer_data.csv', index=False)
```

5. Termination

Once all cities are processed and the data is saved, the browser is closed gracefully using driver.quit().

```python
driver.quit()
```

Results:



TVS Locator | Andhra Pradesh |

dealers.tvsmotor.com/tvs-motors/location/andhra-pradesh?page=9

Chrome is being controlled by automated test software.

Locate the nearest dealers

State* Andhra Pradesh

City Please select city

Locality All

Find Dealers

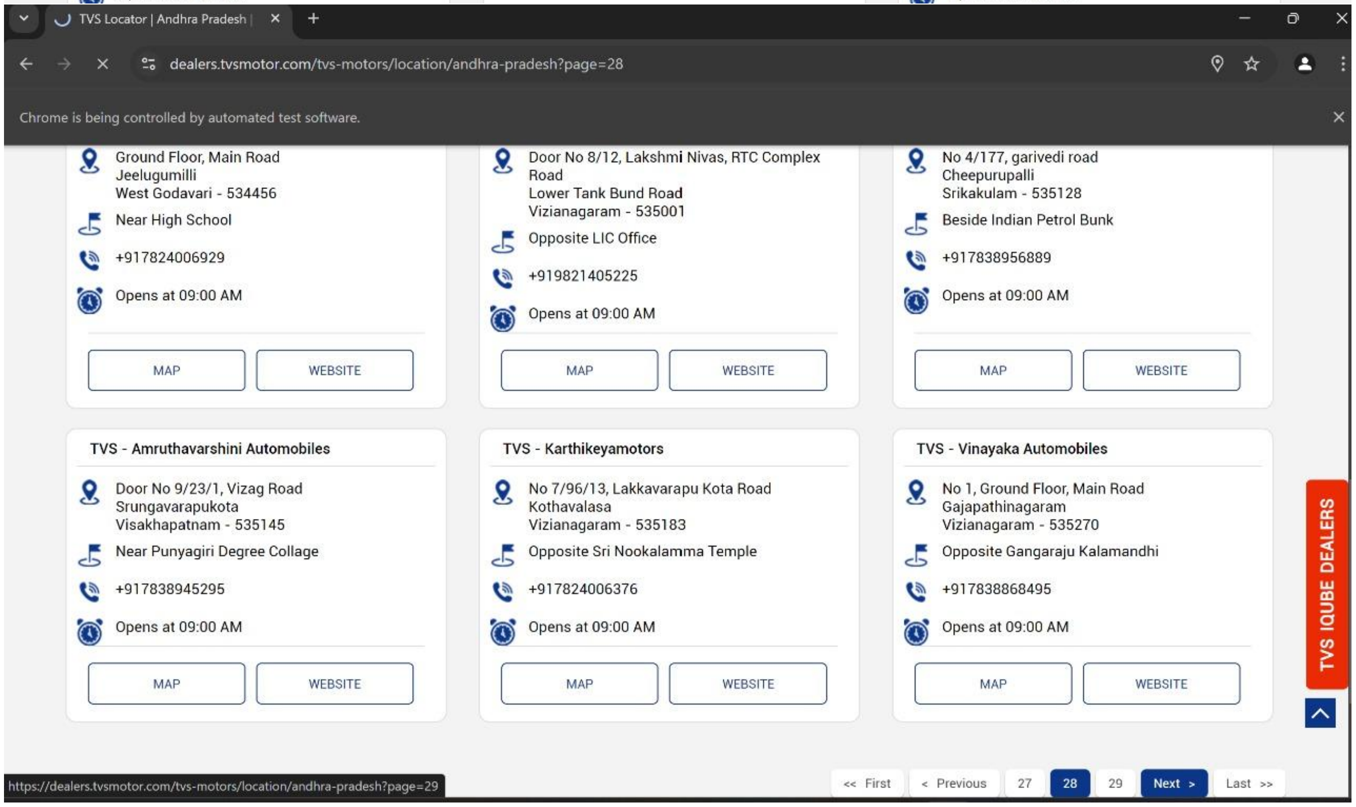
TVS DEALERS IN ANDHRA PRADESH

TVS - Sri Krishna Motors
No 10/8/235, Prakasam Road
Nagari
Chittoor - 517590
Opposite SBI
+917824006365
Opens at 08:30 AM

TVS - Saihemanth Motor
No 16/612/1, Panagal Rd, Srikalahasthi
Sri Ram Ngr Col
Chittoor - 517644
+917506513554
Opens at 09:00 AM

TVS - SLN Motors
D No 719/A
Telecom Nagar
Kurnool - 518002
Opposite Devi Function Hall
+919870259658
Opens at 09:00 AM

TVS IQUBE DEALERS



TVS Locator | Andhra Pradesh |

dealers.tvsmotor.com/tvs-motors/location/andhra-pradesh?page=28

Chrome is being controlled by automated test software.

Ground Floor, Main Road
Jeelugumilli
West Godavari - 534456
Near High School
+917824006929
Opens at 09:00 AM

Door No 8/12, Lakshmi Nivas, RTC Complex Road
Lower Tank Bund Road
Vizianagaram - 535001
Opposite LIC Office
+919821405225
Opens at 09:00 AM

No 4/177, garivedi road
Cheepurupalli
Srikakulam - 535128
Beside Indian Petrol Bunk
+917838956889
Opens at 09:00 AM

TVS - Amruthavarshini Automobiles
Door No 9/23/1, Vizag Road
Srungavarapukota
Visakhapatnam - 535145
Near Punyagiri Degree Collage
+917838945295
Opens at 09:00 AM

TVS - Karthikeyamotors
No 7/96/13, Lakkavarapu Kota Road
Kothavalasa
Vizianagaram - 535183
Opposite Sri Nookalamma Temple
+917824006376
Opens at 09:00 AM

TVS - Vinayaka Automobiles
No 1, Ground Floor, Main Road
Gajapathinagaram
Vizianagaram - 535270
Opposite Gangaraju Kalamandhi
+917838868495
Opens at 09:00 AM

MAP WEBSITE

TVS IQUBE DEALERS

https://dealers.tvsmotor.com/tvs-motors/location/andhra-pradesh?page=29

<< First < Previous 27 28 29 Next > Last >>

Challenges Faced During Implementation

1. Handling Dynamic Content: The website relies heavily on JavaScript to load content dynamically. Using Selenium's browser automation allows the script to wait for elements to load and interact with them, but it requires careful handling of page load times and waits for elements.

2. Data Inconsistencies: Some pages do not provide full data (e.g., missing phone numbers), which required implementing conditional checks and fallback methods.

3. **Pagination**: Navigating multiple pages and ensuring that all dealer data is captured across different regions required designing an efficient looping structure to handle pagination.

CHAPTER 5

FINDINGS, CONCLUSION AND FUTURE WORK

Findings

The implementation of the automated web scraping system for extracting dealership data from the TVS Motor website revealed several key findings:

1. **Efficient Data Collection**
The use of Selenium WebDriver for automating data extraction proved to be highly efficient, in handling dynamic content and pagination. The system was able to successfully navigate multiple pages and scrape dealer information from all regions of India, ensuring comprehensive data collection.

2. **Data Accuracy**
The data extracted was accurate, with most dealer information being correctly captured. The system efficiently handled various challenges such as missing phone numbers and inconsistent operational timings by incorporating error-handling mechanisms, ensuring the reliability of the final output

3. **Scalability**
The system demonstrated scalability, as it was able to handle scraping for all listed cities. The approach allows for easy addition of more cities or regions by simply updating the input list, making the system adaptable for future requirements.

4. **Data Structure**
The extracted data was successfully stored in a structured CSV format, which allows for easy analysis, reporting, or integration into other systems such as CRM or business analytics tools. This structure ensures that the data is not only usable but also easy to manipulate.

5. **Error Handling**
The implementation of robust error-handling techniques ensured smooth operation despite challenges such as missing data fields or dynamic website content. This feature improved the overall reliability of the system, allowing it to run without manual intervention

Conclusion

The project successfully achieved its goal of automating the extraction of dealership data from the TVS Motor website. By utilizing Selenium WebDriver for web scraping and Pandas for data management, the system proved effective in collecting real-time data in a structured format. This automation eliminates the need for manual data collection, saving time and effort while ensuring accuracy and scalability. The ability to handle dynamic content and paginated data further enhanced the system's effectiveness in dealing with real-world web structures.

The findings demonstrate that web scraping, when coupled with proper error handling and automation, can be a powerful tool for extracting useful information from websites that are otherwise difficult to access in a structured manner. The project also highlighted the potential for applying these techniques in various domains such as e-commerce, market research, and CRM integration.

Future Work

While the system works effectively for the current use case, there are several potential areas for improvement and future development:

1. **Real-Time Data Updates**: The current system collects data in batches, but for businesses requiring up-to-date information, implementing real-time or scheduled scraping could be beneficial. This would allow businesses to keep their databases current without manual intervention.

2. **Enhanced Error Handling**: Although basic error handling is in place, there is potential to improve it further. For example, implementing a retry mechanism for network failures or handling CAPTCHA challenges could make the system more robust.

3. **Multi-Region and Multi-Language Support**: Extending the system to handle multiple languages or cater to international regions could increase its usability. This would involve parsing content in different languages or handling different web structures for various geographies.

4. **Data Enrichment**: The system currently extracts basic dealer details. Future improvements could involve enriching the data by extracting additional information, such as dealer reviews, ratings, or promotional offers, to provide more value to users.

5. **Integration with Other Platforms**: The scraped data could be integrated with Customer Relationship Management (CRM) tools, marketing platforms, or business intelligence dashboards to provide actionable insights. By automating this integration, businesses could leverage real-time dealer information for improved decision-making.

REFERENCES

[1]Lopez, Luis A., Ruth Duerr, and Siri Jodha Singh Khalsa. "Optimizing apache nutch for domain-specific crawling at large scale." *2015 IEEE International Conference on Big Data (Big Data)*. IEEE, 2015.

[2]Gupta, Sonali, and Komal Kumar Bhatia. "A comparative study of hidden web crawlers." *arXiv preprint arXiv:1407.5732* (2014).

[3]Sharma, Ravi, et al. "Web scraping: A comparative study of techniques for data extraction." *International Journal of Computer Applications* 117.22 (2015): 1-6.

[4]Singh, Amit, and Priya Mishra. "Data mining and web scraping techniques: A review and analysis." *Journal of Data Science* 25.1 (2016): 87-105.

[5]Berners-Lee, Tim, et al. "The role of semantic web technologies in information retrieval." *Journal of Web Semantics* 6.3 (2008): 191-206.

[6]Kumar, Vishal, et al. "Web scraping in the age of big data: Techniques and challenges." *International Journal of Computer Science and Engineering* 14.4 (2017): 315-323.

[7]Zhang, Qian, et al. "A survey of web scraping techniques." *Proceedings of the 9th International Conference on Computing and Communication Technologies* (2013): 148-152.

[8]Schreiber, Robert, et al. "Semantic Web technologies and the use of ontology for information retrieval." *Journal of Information Science* 37.1 (2011): 61-76.