PROJECT REPORT

PROJECT TITLE: **WEB SCRAPING & AUTOMATION** **(PYTHON PROJECT)**

NAME: **ISHIKA PARUA** **PROJECT DATE:** **2024**

PROJECT LINK: https://github.com/Ishika002/WEB-SCRAPING-AND-AUTOMATION.git

INTRODUCTION

This project focuses on **web scraping and automation using Beautiful Soup and Selenium**, two powerful Python libraries. Web scraping enables the extraction of data from websites, while automation helps interact with web elements dynamically. **Beautiful Soup** is used for parsing and extracting structured data from static HTML pages, making it ideal for scraping well-structured websites. **Selenium**, on the other hand, is employed to handle dynamic web content, enabling automation of tasks like form submission, page navigation, and handling JavaScript-rendered elements. The project demonstrates how to efficiently **scrape data, automate repetitive tasks, and store the extracted information for further analysis, making it useful for data collection, market research, and process automation.**

DATA & METHODOLOGY

- **Data Source:** Websites with structured or dynamic content (e.g., e-commerce, news, or financial data).
- **Tools & Libraries:** Python, Beautiful Soup (for parsing static HTML), Selenium (for interacting with dynamic web pages), Pandas (for data processing).
- **Data Extraction:** Beautiful Soup for scraping structured HTML; Selenium for handling JavaScript-rendered content and automating actions like clicking and scrolling.
- **Data Processing:** Cleaning and structuring extracted data using Pandas, handling missing values, and converting data into usable formats (CSV, JSON, or databases).
- **Automation:** Using Selenium for tasks like login automation, periodic data updates, and interaction with form fields and buttons.

KEY INSIGHTS & FINDINGS

- **Efficient Data Extraction:** Automated retrieval of structured and dynamic web data, reducing manual effort.
- **Handling Dynamic Content:** Selenium successfully interacts with JavaScript-based elements, enabling seamless scraping of modern websites.
- **Data Accuracy & Cleaning:** Extracted data requires preprocessing to handle missing values, duplicates, and inconsistencies.
- **Automation Benefits:** Automating repetitive tasks like form submissions, login processes, and scheduled data updates enhances efficiency.
- **Challenges & Limitations:** Websites with anti-scraping mechanisms (CAPTCHAs, dynamic loading) require advanced handling techniques like proxies and headless browsing.

CONCLUSION & IMPACT

This project demonstrates the power of web scraping and automation using Beautiful Soup and Selenium to efficiently extract and process data from websites. By automating repetitive tasks, it enhances productivity and reduces manual effort in data collection. The insights gained from structured data can be leveraged for market analysis, trend identification, and decision-making. Future improvements could include integrating APIs, handling CAPTCHA challenges more effectively, and optimizing performance for large-scale data extraction.