**Competitor Scraper project**

**Raoof Aranji valappil**

# Executive summary

In the increasingly competitive business environment, staying informed about market trends and competitor activities is essential for strategic decision-making and maintaining a competitive edge. This project aims to develop an advanced web scraping tool to gather comprehensive data on competitors from the GOV.UK Digital Marketplace. The primary goal is to extract and analyze critical information such as company names, revenues, types of services offered, number of employees, and CEO names. This information will enable businesses to gain valuable insights into their competitive landscape and make informed strategic decisions. The project will leverage Scrapy, a robust and efficient web crawling and scraping framework written in Python, known for its high performance and scalability. Visual Studio Code will be used as the development environment, providing an intuitive and powerful platform for writing and managing the project code. To ensure the scraped data remains current and accurate, Mage will be utilized to automate updates to a MongoDB database. This database will act as a centralized repository for the collected data, allowing for easy manipulation and analysis. For visualization and analysis, Power BI will be employed to create dynamic and insightful dashboards, highlighting trends, patterns, and actionable insights from the data. Also, the project will include developing a user-friendly interface that allows users to customize their data extraction preferences. This feature will enable users to select specific types of data and target websites, enhancing the tool’s flexibility and usability. The successful implementation of this project will result in a powerful tool for competitor analysis, streamlining the process of data collection and providing businesses with valuable insights. This tool will support strategic planning and decision-making, ultimately helping businesses to stay competitive in a rapidly evolving market. By integrating advanced technologies and emphasizing user-centric design, this project underscores the critical role of data science in modern business strategy. The outcome will be a comprehensive solution that not only simplifies the data gathering process but also delivers meaningful insights that can drive business growth and competitiveness.

# Table of contents

[Executive summary 1](#_Toc819529536)

[Table of contents 2](#_Toc1335900034)

[Introduction 3](#_Toc1937268576)

[Planning and Research 4](#_Toc497572880)

[Concept of Project Ideas 5](#_Toc1583285503)

[Gathering Requirements 6](#_Toc957841131)

[Risk Analysis 6](#_Toc559752123)

[Initial Evaluation 8](#_Toc1102250964)

[Backend 9](#_Toc1197102769)

[Testing web and email scrappers 10](#_Toc155469877)

[Evaluation of scrappers 11](#_Toc945970447)

[Implementation of final scrappers 12](#_Toc132842149)

[Evaluation of final scrappers 13](#_Toc1996153524)

[Conclusion 15](#_Toc501512887)

# Introduction

In today's highly competitive business environment, understanding the competitive landscape is crucial for making informed strategic decisions. This project aims to develop a sophisticated web scraping tool designed to gather and analyze key information about competitors from the GOV.UK Digital Marketplace. The primary objective is to extract vital data points such as company names, revenues, types of services offered, number of employees, and CEO names. This information will be used to gain insights into market trends, identify business opportunities, and enhance strategic planning.

The project will employ advanced web scraping techniques using Scrapy, a powerful and efficient web crawling and scraping framework written in Python. Scrapy's ability to handle multiple requests asynchronously makes it an ideal choice for extracting large volumes of data quickly and accurately. Visual Studio Code will be utilized as the development environment, providing a robust platform for coding, debugging, and managing the project. To ensure the scraped data is up-to-date and accurately reflects the current market conditions, Mage will be used to automate updates to a MongoDB database. This database will serve as the central repository for all scraped data, enabling efficient data manipulation and analysis. Power BI will be employed for data visualization, creating dynamic dashboards and visualizations that facilitate easy interpretation of the data. These visualizations will help identify trends, patterns, and insights, providing valuable input for strategic decision-making. Additionally, the project will feature a user-friendly interface that allows users to customize their data extraction preferences. Users will be able to select specific data types and target websites, enhancing the flexibility and usability of the tool.

By integrating these advanced technologies, the project aims to deliver a comprehensive solution for competitor analysis. The final outcome will not only streamline the data gathering process but also provide actionable insights that can drive business growth and competitiveness. This project underscores the importance of leveraging modern data science tools to stay ahead in a rapidly evolving market landscape.

# Planning and Research

For the planning and research phase of the project, I focused on understanding the scope and requirements thoroughly. The objective is to scrape key information from the GOV.UK Digital Marketplace, including company names, revenues, types of services offered, number of employees, and CEO names. This data is essential for analyzing the competitive landscape. To achieve this, I conducted extensive research on two familiar web scraping libraries such as Selenium, and Scrapy. Selenium can automate web browsers and interact with dynamic web pages but tends to be slower due to its browser automation. Scrapy, on the other hand, is a high-level web scraping and crawling framework designed for performance and scalability. It handles multiple requests asynchronously and includes built-in features like request scheduling, automatic retries, and support for pipelines and middleware. After evaluating these options, I chose Scrapy and Python for the project due to their efficiency, performance, and scalability.

For the development environment, I selected Visual Studio Code because it is a robust and familiar platform for coding, debugging, and managing Python projects. To automate the updates of scraped data into a MongoDB database, I decided to use Mage. This automation is crucial for maintaining the data's integrity and timeliness without manual intervention. For data visualization and analysis, I will use Power BI. This tool allows for the creation of detailed dashboards and visualizations, enabling the identification of trends, patterns, and insights from the scraped data. Additionally, an interface will be designed to allow users to select the type of data they want and specify the websites from which to extract this information. This user-centric approach ensures flexibility and ease of use, enhancing the overall functionality of the project.

# Concept of Project Ideas

I conducted thorough research to understand the scope and objectives of the project. The main goal is to scrape essential information from the GOV.UK Digital Marketplace. The key data points to be extracted include the name of the company, their revenue, the type of services they offer, the number of employees, and the CEO's name. This information is crucial for analyzing the competitive landscape and making informed business decisions. For this project, I have chosen to use Scrapy, a high-level and efficient web crawling and scraping framework written in Python. Scrapy is particularly suitable for this task due to its ability to handle multiple requests asynchronously, which makes the data extraction process fast and efficient. It also includes built-in features such as request scheduling and automatic retries, which enhance the robustness and reliability of the scraping process. I will develop and manage the project code using Visual Studio Code, a platform I am well-versed with. Its features for efficient coding, debugging, and project management will streamline the development process. To ensure the scraped data remains current and accurate, I will use Mage to automate updates to a MongoDB database. This automation will reduce manual intervention and keep the data repository up to date, allowing for efficient data manipulation and analysis. For data visualization and dashboarding, I will utilize Power BI. This powerful tool will help create insightful visualizations and dashboards, enabling easy analysis of trends, patterns, and insights from the scraped data. The data will be filtered and presented in formats conducive to analysis, ensuring its practical utility. Furthermore, I will design an interface that allows users to select the type of data they want and specify the websites from which to extract this information. This user-centric approach will enhance the usability and flexibility of the project, ensuring it meets the varied needs of its users effectively.

In summary, this project aims to create a comprehensive system for scraping and analyzing competitor data from the GOV.UK Digital Marketplace. By leveraging Scrapy for data extraction, Mage for automation, MongoDB for data storage, and Power BI for visualization, the project will provide valuable insights into the competitive landscape, facilitating informed business decisions. The user-friendly interface will further ensure the project’s practical application and accessibility.

# Gathering Requirements

Python was chosen as the programming language due to its versatility and extensive library ecosystem, ideal for web scraping. Visual Studio Code was selected as the code editor for its lightweight yet powerful features, suitable for project management and development across multiple platforms. Extensive research identified Scrapy, and Selenium as the most suitable libraries for data mining and web scraping tasks. The project will implement algorithms for HTML parsing, data cleaning, storage, and analysis to effectively process the extracted data.

# Risk Analysis

The project encountered significant risks during the initial phase, including delays due to illness, extended timeline for requirement identification, and library compatibility issues. These challenges necessitated adjustments to the project timeline and milestones, highlighting the importance of flexibility and proactive risk management. By addressing these risks and documenting the mitigation strategies, a more resilient project plan was developed to overcome hurdles effectively.

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk** | **Potential Impact** | **Mitigation Strategies** | **Status** |
| **Severe illness and fever** | Delay in project kickoff and extended timeline, resulting in initial delays in completing tasks. | Adjust project timeline, allocate additional time for catch-up sessions, and ensure flexibility in project planning. | Addressed and Resolved |
| **Delayed requirement identification** | Extended time to identify project requirements, impacting overall schedule and progress. | Conduct detailed planning sessions, involve stakeholders early, and set realistic deadlines for requirement gathering. | Addressed and Resolved |
| **Incompatible libraries** | Initially chosen libraries proved unsuitable for data scraping, causing delays in implementation and testing phases. | Research and test alternative libraries, choose the best-performing library (Scrapy), and re-evaluate project tools regularly. | Addressed and Resolved |
| **Timeline adjustments** | Need for timeline adjustments to accommodate unexpected delays and challenges. | Establish realistic milestones, maintain a flexible project schedule, and continuously monitor progress. | Managed Continuously |
| **Single developer workload** | High workload on a single developer, risking burnout and further delays. | Implement efficient time management strategies, seek support from peers or mentors, and prioritize tasks effectively. | Ongoing Management |
| **Data integrity and validation** | Risks of data inconsistencies and inaccuracies during scraping and storage. | Implement robust data validation and cleaning processes, use reliable storage solutions (MongoDB), and conduct regular checks. | Addressed and Resolved |
| **Technology stack learning curve** | Learning new libraries and tools may take longer than expected, impacting productivity. | Allocate time for learning and experimentation, access tutorials and documentation, and apply best practices in implementation. | Managed Continuously |

By proactively identifying and managing these risks, the project plan was adjusted to accommodate delays and challenges. Detailed documentation and a focus on flexibility ensured that the project remained on track and resilient to future hurdles. The adoption of suitable technologies and efficient time management strategies contributed to the project's success.

# Initial Evaluation

The objective of this task is to assess the project's feasibility based on comprehensive research and analysis conducted in previous tasks. The technological stack, attributes for data extraction, and filters to refine data have been identified to ensure clarity and alignment within the team. By the end of this phase, a decision will be made on whether to proceed with the project, considering its viability and potential impact. Should the project be deemed unfeasible, a conclusion will be reached to cease development.

# Backend

During the backend development phase, the project aimed to identify, evaluate, and select the most suitable web scraping library for extracting key information from the GOV.UK Digital Marketplace. This involved setting up the development environment using Visual Studio Code and testing libraries such as Selenium, and Scrapy, to understand their capabilities and limitations. Basic web scrapers were developed for each library to extract sample data, focusing on company names, revenue, services offered, number of employees, and CEO names. Performance metrics, including speed, accuracy, and resource consumption, were measured to evaluate the ease of use and flexibility of each library. Following comprehensive testing, Scrapy emerged as the optimal choice due to its superior performance, scalability, and built-in features. MongoDB was selected as the data storage solution, with Mage utilized to automate updates to the database. Integration with Power BI enabled the creation of visualizations and dashboards for insightful data analysis, supporting effective decision-making. This backend development process ensures efficient extraction, storage, and analysis of key information from the GOV.UK Digital Marketplace, aligning with project objectives seamlessly.

# Testing web and email scrappers

This part of the project focuses on testing and evaluating various web scraping libraries to determine the most efficient tool for extracting data from the GOV.UK Digital Marketplace. The primary goal is to measure the performance of each library, identifying the one that offers the best speed, accuracy, and reliability for large-scale data scraping tasks. We evaluated Selenium, and Scrapy, testing each for its ability to extract key information such as company names, revenue, services offered, number of employees, and CEO names. Scrapy emerged as the best choice over Selenium due to its asynchronous and concurrent scraping capabilities, allowing it to make multiple requests simultaneously, thus being faster and more efficient. Scrapy's scalability ensures it can handle growing project demands by distributing tasks across multiple processes or machines. Additionally, Scrapy’s built-in features like request scheduling, automatic retries, and support for robots.txt parsing make the scraping process robust and reliable. Its powerful tools for structured data extraction. Scrapy’s support for pipelines and middleware allows efficient data processing, cleaning, and transformation before storage. With a large and active community, Scrapy offers extensive documentation and third-party libraries, providing valuable resources for troubleshooting and extending its functionality. Performance metrics showed that Scrapy took 1 minute and 49 seconds to scrape company details and services, significantly outperforming Selenium, which took 127 minutes and 37 seconds. Based on these findings, Scrapy was selected for its superior speed and scalability. The scraped data is stored in a MongoDB database, with Mage automating updates to ensure accuracy. The data is then integrated with Power BI to create insightful visualizations and dashboards, facilitating the analysis of trends and patterns. This comprehensive approach ensures the project's success in efficiently and effectively extracting and analyzing data from the GOV.UK Digital Marketplace.

# [Evaluation of scrappers](https://github.com/NachoDAS/Competitor-Scrapper/issues/10)

To determine the most suitable web scraper for the project, I conducted a thorough evaluation of two popular libraries: Selenium, and Scrapy. Selenium excels at interacting with dynamic web pages and can simulate user interactions, making it versatile for browser automation and web testing. However, its performance is significantly slower as it operates by controlling a web browser. In contrast, Scrapy is built for efficient web scraping, capable of handling multiple requests asynchronously, making it much faster than Selenium. Scrapy's built-in features for handling requests, extracting data, and managing common scraping tasks like retries and user-agent rotation make it suitable for large-scale scraping projects. Additionally, Scrapy has a large and active community providing extensive resources and support.

The performance metrics highlighted Scrapy's advantages. Scrapy demonstrated the fastest scraping speed, followed by Selenium being the slowest. While these two libraries provided accurate data extraction, Scrapy's built-in tools streamlined the process more efficiently. Its asynchronous processing resulted in lower resource consumption compared to Selenium's browser-based scraping. Based on these outcomes, Scrapy was identified as the most suitable tool for the project. Scrapy's high performance, extensive built-in features, and scalability make it ideal for handling large-scale data extraction. Its ability to manage multiple requests asynchronously ensures faster and more efficient data scraping. The next steps involve developing the web scraper using Scrapy, optimizing it for maximum performance and accuracy, and integrating the scraped data into the MongoDB database using Mage. By selecting Scrapy, I am confident that the project goals can be achieved effectively and efficiently.

# Implementation of final scrappers

During the task focused on cleaning and optimizing the best web scraper, several important outcomes were achieved to ensure its efficient and seamless integration into the backend of the project. The selected web scraper, Scrapy, underwent code refinement and optimization to remove any redundancies, improving both readability and maintainability. By enhancing the efficiency of data extraction processes, the performance of the scraper was significantly improved, reducing the time required for scraping. Comprehensive error handling mechanisms were implemented to manage potential issues such as network failures, timeouts, and unexpected changes in website structure. Logging functionality was added to capture detailed error reports, facilitating easier debugging and maintenance. Data cleaning processes were integrated to remove duplicates, handle missing values, and standardize data formats, ensuring the extracted data is clean and reliable. Additionally, data integrity was validated to confirm that the scraper accurately extracts the required information from the target websites.

The scraper’s architecture was reorganized into modular components, allowing for easier updates and scalability. This modular approach also prepared the scraper for potential future enhancements, such as adding new data fields or integrating with additional websites. Comprehensive documentation was created detailing the scraper’s design, implementation, and optimization steps, ensuring clarity for future developers. Coding best practices and guidelines were established to maintain high code quality and facilitate seamless collaboration. Following these optimizations, the scraper was fully integrated into the backend of the project, operating as a core component of the data extraction process. Automated schedules were set up for the scraper to run at regular intervals, ensuring the data remains up-to-date and relevant.

To ensure the ongoing performance and reliability of the scraper, continuous monitoring tools were implemented to track its performance and promptly address any issues that arise. Regular maintenance checks were scheduled to update the scraper and ensure it adapts to any changes in the target websites’ structures. The cleaned and validated data was stored in a MongoDB database, making it easily accessible for analysis and visualization. Power BI was used to create dynamic visualizations and dashboards, turning the scraped data into actionable insights. Plans for scalability were put in place, designing the scraper to handle increased data volumes and the addition of new target websites. Potential future enhancements were explored, such as incorporating machine learning models to predict trends and patterns based on the scraped data. These steps ensure that the web scraper remains efficient, reliable, and scalable, meeting the project’s data extraction needs effectively.

# Evaluation of final scrappers

The evaluation of the backend web scrapers developed using Scrapy and Selenium provided a comprehensive analysis of their performance, focusing on accuracy, precision, and speed of data extraction from the specified website. The key findings from the performance tests were significant. Scrapy demonstrated remarkable efficiency by completing the data scraping task in just 1 minute and 49 seconds. In contrast, Selenium took a considerably longer time, requiring 127 minutes and 29 seconds to perform the same task. This stark contrast in performance highlights the superior speed and efficiency of Scrapy, particularly for large-scale data scraping projects. The results clearly indicate that Scrapy is highly efficient and well-suited for handling extensive data extraction requirements, making it the preferred choice for projects demanding high performance.

The substantial difference in scraping times between Scrapy and Selenium underscores Scrapy's capability to handle large volumes of data quickly and accurately. This efficiency is critical for large-scale projects where timely data extraction is crucial. Adopting Scrapy as the primary web scraping tool ensures not only faster data extraction but also improved productivity and resource utilization. Furthermore, Scrapy's robust framework allows for additional optimizations and enhancements, which can be explored to maximize its benefits. By leveraging Scrapy's advanced features and performance, the project can achieve greater efficiency and accuracy in data extraction, thereby meeting the project's goals effectively and efficiently. This evaluation reaffirms Scrapy's position as the optimal tool for large-scale web scraping tasks.

# Conclusion

The Competitor Scraper project aims to provide valuable insights into competitor data for strategic decision-making in competitive market analysis. Through the utilization of Python and specialized web scraping libraries, the project seeks to automate the process of data extraction, analysis, and visualization. By addressing potential risks and implementing mitigation strategies, the project aims to deliver a robust and effective solution for competitive market research under the Distributed Analysis company.