**Company Director Extractor using NLP and Web Scraping**

This project is a fully automated system developed using Python that extracts the names of directors, board members, and key personnel from official company websites. The tool integrates **web scraping** and **Natural Language Processing (NLP)** using the spaCy library to identify and extract human names from unstructured text on webpages.

The purpose of this project is to support analysts, researchers, or data engineers who require structured information about corporate leadership but face the challenge of manually browsing and parsing hundreds of company websites. With this solution, company details and director names can be retrieved accurately, efficiently, and stored in a reusable format.

### ****Key Features:****

* **Web Scraping with Requests & BeautifulSoup:** Extract raw HTML and clean visible text from web pages.
* **NLP with spaCy (en\_core\_web\_lg):** Identify names using Named Entity Recognition (NER).
* **Text Cleaning:** Remove irrelevant content like scripts and styles for better accuracy.
* **Duplicate Handling:** Remove repeated names using sets.
* **Error Handling:** Gracefully manages network failures, bad URLs, and empty pages.
* **Retry Mechanism:** Allows reprocessing of previously failed websites.
* **CSV Export:** Stores both successful and failed extractions for transparency and auditability.

### ****Technologies Used:****

* **Language:** Python 3
* **Libraries:**
  + requests – For fetching web content
  + BeautifulSoup – For parsing HTML
  + spaCy – For NLP and NER
  + pandas – For data storage and manipulation
  + time – To manage request pacing

### ****Input/Output:****

* **Input CSV:** Contains Company Name and URL fields.
* **Output CSVs:**
  + directors\_extracted\_success.csv – With extracted names
  + directors\_extracted\_failed.csv – For unresolved sites

Example structure:

| **Company Name** | **URL** |
| --- | --- |
| ABC Ltd | <https://www.abcltd.com> |

### ****How It Works (Workflow):****

1. **Read the input files** (df\_success, df\_failed)
2. **For each URL:**
   * Download HTML content using requests
   * Clean and extract visible text via BeautifulSoup
   * Pass text to spaCy NLP model to detect names
   * Clean and filter those names (at least 2 words, remove irrelevant roles)
3. **Store**:
   * Append names to df\_success if found
   * Else append to df\_failed for future retry

### ****Results:****

* Extracted dozens to hundreds of directors from multiple domains
* Significantly improved extraction success after adding retries
* Reduced manual work and increased reproducibility

### ****Limitations:****

* May fail on sites using heavy JavaScript rendering
* Cannot distinguish roles unless explicitly written next to names
* NER may include false positives (e.g., location names or unusual titles)

**Future Improvements:**

* Use **Selenium** or **Playwright** to handle JavaScript-heavy websites.
* Add **role classification** to label titles like CEO, CFO, etc.
* Deploy as an **API** or **Flask-based web app** for interactive use.
* Integrate with a **database** or **dashboard** for enterprise-grade analysis.
* **Enhance accuracy** by combining rule-based name filtering with spaCy NER.
* Implement **multi-language support** for global company websites.
* **Support PDF parsing**: Many companies publish leadership info in annual reports or PDF brochures. Integrating a PDF extractor (e.g., PyMuPDF, pdfminer.six, or pdfplumber) would allow the system to fetch and extract director names from PDF files linked on websites.

### ****Conclusion:****

This project successfully demonstrates how NLP and web scraping can be combined to extract structured data (director names) from unstructured sources (websites). With a modular architecture and proper logging of failures, the system is both scalable and maintainable. It lays the groundwork for more advanced corporate intelligence systems in the future.