

Matteo Alberici - Information Retrieval 2021

---

## CNE - Companies Search Engine

---

Project Report

### Contents

<b>1. Introduction</b>	<b>2</b>
<b>2. Web Crawling with Scrapy</b>	<b>2</b>
2.1. Forbes . . . . .	3
2.2. Wikipedia . . . . .	4
2.3. Indeed . . . . .	5
2.4. Britannica . . . . .	6
2.5. Displaying Results . . . . .	7
<b>3. Indexing with Solr</b>	<b>7</b>
<b>4. Website with HTML, CSS, and JavaScript</b>	<b>8</b>
<b>5. User Evaluation</b>	<b>10</b>
<b>6. Conclusion</b>	<b>11</b>

# 1. Introduction

For the project of the Information Retrieval course, I was requested to develop a search engine which gathers a large collection of companies and the information associated with them, enabling to search this collection. The results should have been presented to the user in a tabular format, so that many companies could be seen at the same time, with all the information related to them.

Information Retrieval  
Academic Year 2021-22  
**Course Project N.02: Company Search Engine**  
Prof. Fabio Crestani

For this assignment you will work alone to implement a working prototype of an information retrieval system for a specific task and user needs. The goal of this assignment is to apply the concepts and tools you have learned in the class to a practical, real world, application.

**Description**

Company Search Engine: a user wants to search for a company on websites such as GlassDoor or Indeed. You need to build a system that gathers a large collection of companies and the information associated with them and enable to search this collection. The system has to provide the best interface for searching, browsing, and presentation of the data to the user. The system should also have the following additional feature:

- Results presentation: results should be presented in a tabular format, so that many companies could be seen at the same time. Each table cell should contain the following information about a company: name, headquarters, industry, rating (if available) and size.

Figure 1: Project description

The project can be divided into four sections: the web crawling part, which has been performed with [Scrapy](#), the indexing part, performed with [Solr](#), the user interface, which was developed using HTML, CSS, and Javascript, and finally the user evaluation part.

## 2. Web Crawling with Scrapy

Before starting, let me introduce some definitions for concepts I will talk about in the following section: with web crawling, we mean the process of automatically downloading a web page's data, extracting the hyperlinks it contains and following them, while with web scraping we refer to the process of automatically downloading a Web page's content and extracting specific information from it.

First of all, I had to gather a large amount of data concerning companies from different websites. The whole crawling part of the project was developed using python and Scrapy. I chose four websites to crawl in order to get a collection which would have been large enough. The selected websites were the following ones:

- [Forbes](#)
- [Wikipedia](#)
- [Indeed](#)
- [Britannica](#)

```

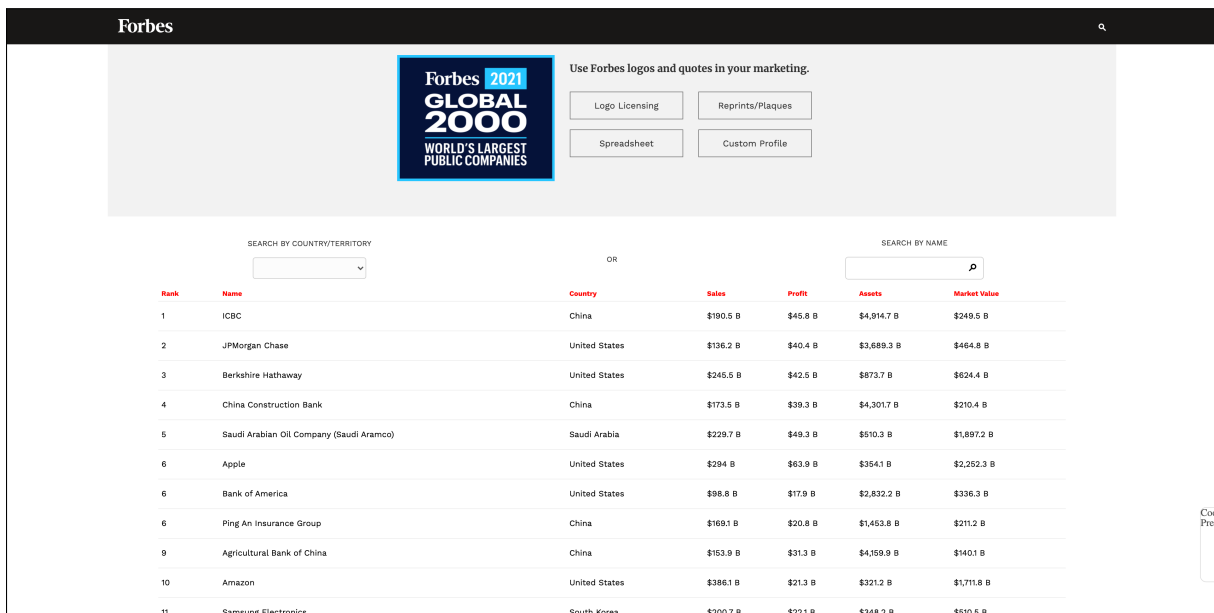
alberma@MBP-di-Albe cse_crawler % tree
.
├── cse_crawler
│   ├── spiders
│   │   ├── init__.py
│   │   ├── britannica.py
│   │   ├── indeed.py
│   │   ├── init__.py
│   │   ├── middlewares.py
│   │   ├── settings.py
│   │   ├── data
│   │   │   ├── britannica.json
│   │   │   ├── indeed.json
│   │   │   ├── items.py
│   │   │   ├── pipelines.py
│   │   └── scrapy.cfg
└── alberma@MBP-di-Albe cse_crawler % _

```

Figure 2: Crawler file tree

## 2.1. Forbes

The first companies list I found was on Forbes. More specifically, it was contained in an article called "*Forbes' Global 2000 list*". It had a lot of data I needed, such as the headquarters and the size of a company.



Forbes 2021 GLOBAL 2000 WORLD'S LARGEST PUBLIC COMPANIES

Use Forbes logos and quotes in your marketing.

Logo Licensing Reprints/Plaques Spreadsheet Custom Profile

SEARCH BY COUNTRY/TERRITORY OR SEARCH BY NAME

Rank	Name	Country	Sales	Profit	Assets	Market Value
1	ICBC	China	\$190.5 B	\$45.8 B	\$4,914.7 B	\$249.5 B
2	JPMorgan Chase	United States	\$136.2 B	\$40.4 B	\$3,689.3 B	\$464.8 B
3	Berkshire Hathaway	United States	\$245.5 B	\$42.5 B	\$873.7 B	\$624.4 B
4	China Construction Bank	China	\$173.5 B	\$39.3 B	\$4,301.7 B	\$210.4 B
5	Saudi Arabian Oil Company (Saudi Aramco)	Saudi Arabia	\$229.7 B	\$49.3 B	\$510.3 B	\$1,897.2 B
6	Apple	United States	\$294 B	\$63.9 B	\$354.1 B	\$2,262.3 B
6	Bank of America	United States	\$98.8 B	\$17.9 B	\$2,832.2 B	\$336.3 B
6	Ping An Insurance Group	China	\$169.1 B	\$20.8 B	\$1,453.8 B	\$211.2 B
9	Agricultural Bank of China	China	\$153.9 B	\$31.3 B	\$4,359.9 B	\$140.1 B
10	Amazon	United States	\$386.1 B	\$21.3 B	\$321.2 B	\$1,711.8 B
11	Samsung Electronics	South Korea	\$200.7 B	\$22.1 B	\$348.2 B	\$510.5 B

Figure 3: Forbes' article page

Unfortunately, it was impossible for me to scrape Forbes's article, since every kind of request I performed in the Scrapy shell while designing my spider gave back a null value. At this point, I decided to try to scrape the Forbes' homepage in order to see if I was getting wrong somewhere in my code implementation or if the problem was a Forbes restriction, but I kept being banned even

with the following settings in the ”*settings.py*” file:

```
# Obey robots.txt rules
ROBOTSTXT_OBEY = True
# Configure maximum concurrent requests performed by Scrapy (default: 16)
CONCURRENT_REQUESTS = 2
```

Finally, I had to remove from my project plan the only website I found with a quite large amount of data.

## 2.2. Wikipedia

Wikipedia has many different lists of companies and corporations, and it is also easy to scrape.

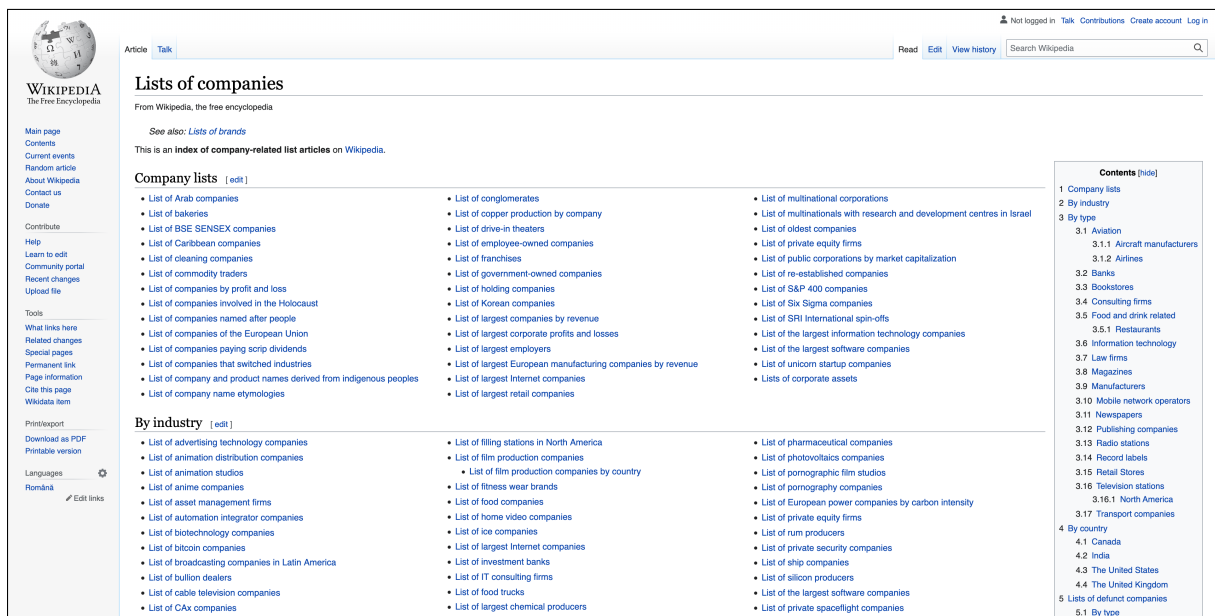


Figure 4: Wikipedia’s companies lists

Here, the problem I encountered was that each list was completely different by the others, with different data and structure, which made impossible for me to design a spider in order to obtain the information I needed. Moreover, many of the lists on Wikipedia had a maximum of 10 companies, which was definitely not enough with respect to the amount of data the project required.



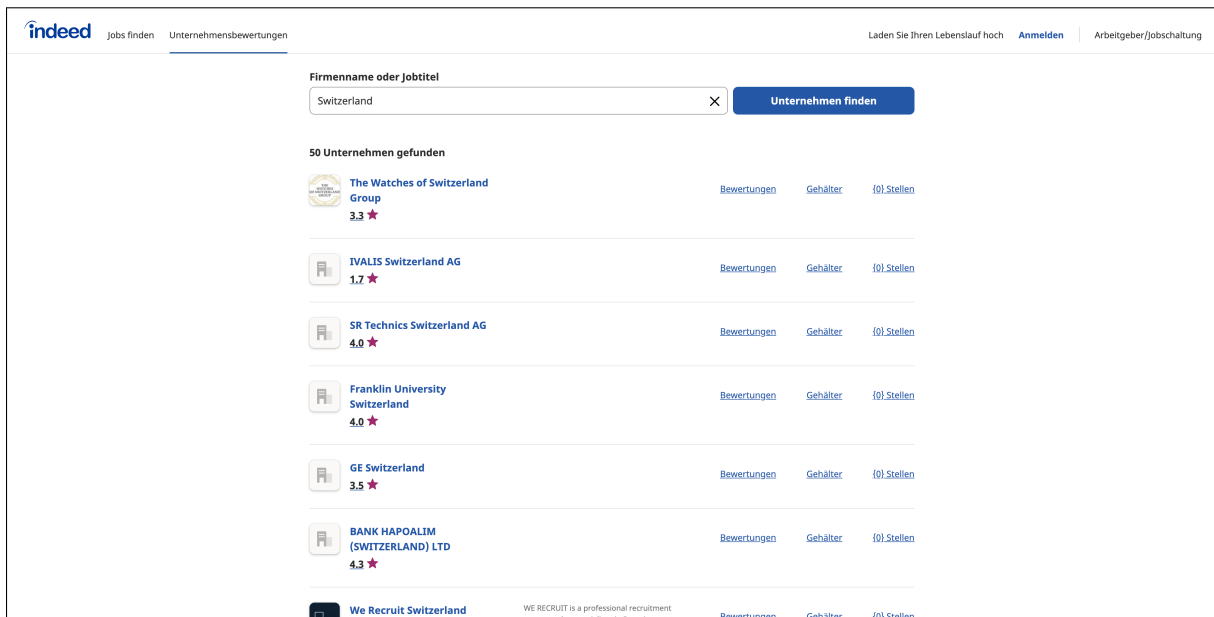


Figure 7: Indeed's list for companies concerning "Switzerland"

Since there is not a "next page" button nor anything to increase the number of results, I ran a spider on different links, each with a different country. The links are the following:

- <https://ch.indeed.com/companies/search?q=switzerland>
- <https://ch.indeed.com/companies/search?q=italy>
- <https://ch.indeed.com/companies/search?q=france>
- <https://ch.indeed.com/companies/search?q=germany>
- <https://ch.indeed.com/companies/search?q=austria>

At the first try, I got banned because I was performing too many requests in a short amount of time, but after a week I got unbanned and I managed to obtain more than 250 results (definitely not enough, but at least I had something I could work with).

## 2.4. Britannica

The last website I tried to scrape was Britannica, where I found a list with many companies gathered depending on the country their headquarters were settled in.

I wrote a new spider for the following link:

<https://www.britannica.com/topic/list-of-corporations-2039518>

Finally, I managed to obtain data from more than 500 different companies.

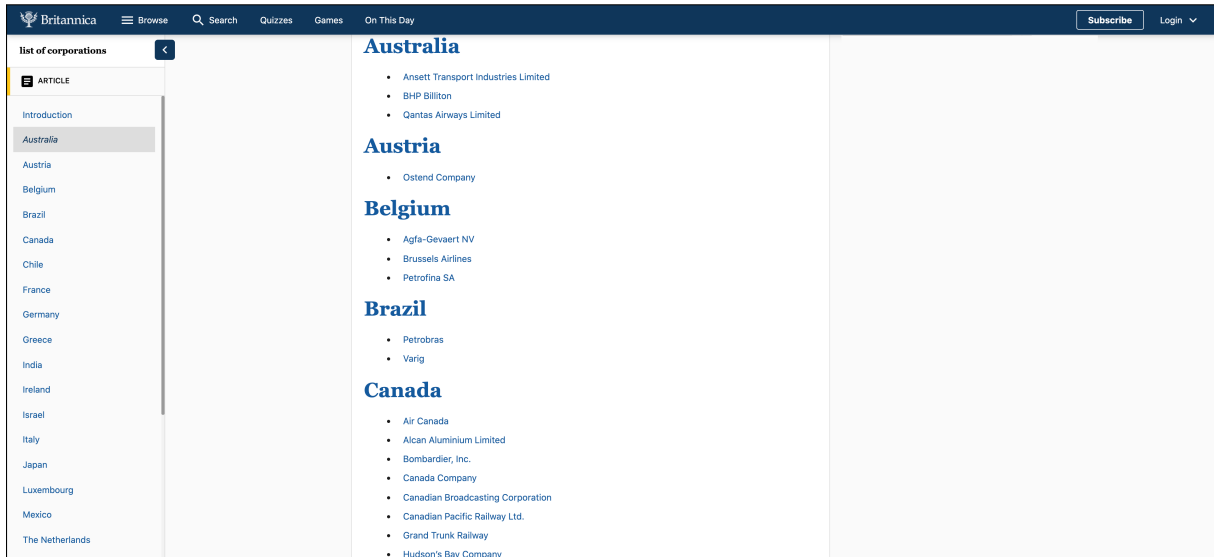


Figure 8: Britannica's list for corporations

## 2.5. Displaying Results

Due to the time spent on understanding Scrapy and due to the large amount of time wasted while trying to scrape Forbes, which was the largest collection I could obtain, I had to work with less than 600 companies, thus I decided to try to gather as much information as I could concerning those companies. Each company has been structured with the following fields:

- Company name
- Company rating (or *N/A*)
- Company Country
- Company size (or *N/A*)
- Company industry

"*N/A*" stands for "*not available*", since each website has different information for each company and fields such as the rating and the size of a company were only displayed in Indeed.

## 3. Indexing with Solr

The all indexing part of the project was developed with Solr, which is an open-source search platform and the most used IR library, and serves as the backend part of the website I developed. First, I created a collection named "*companies*", where each object is structured as follows:

```
{
  "name":["company name"],
  "rating":[company rating],
  "url":["company url"],
  "country":["company country"],
  "size":["company size"],
  "industry":["company industry"],
  "id":"company object id",
  "_version_": object version
}
```

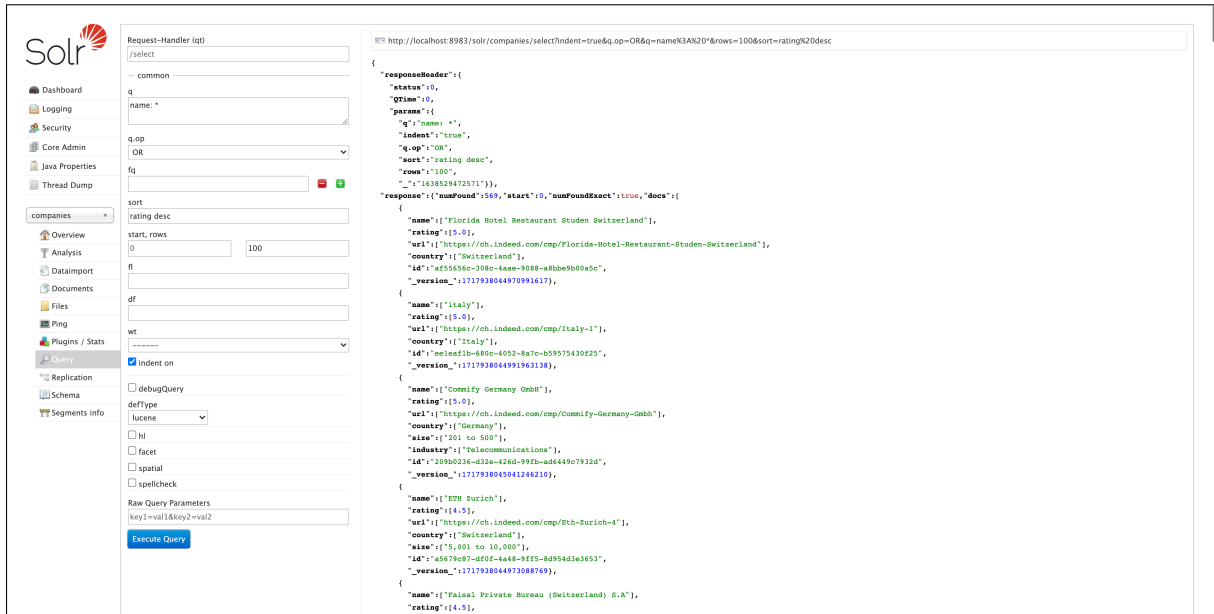


Figure 9: Solar query page

Since in my project a user searches for a specific company, I have to make requests to Solr based on the name typed by the user in the search box. The query I used is structured as follows:

$$http://localhost:8983/solr/companies/select?indent=true\&q.op=OR\&q=name\%3A + name + '*\&rows=100\&sort=rating\%20desc,$$

where:

- *companies* is the collection of companies
- *q=name* is the type of query which is searching for a match with the name given in input
- *name* is the name of the company typed by the user in the search box
- *sort=rating%20desc* is the sorting parameter, meaning that the companies are sorted depending on their rating in decreasing order

After designing the query type, I had to develop a website with a JavaScript file which would have performed a request to Solr in order to retrieve relevant documents from the collection.

## 4. Website with HTML, CSS, and JavaScript

The last part of the developing process of my project consisted in creating a website in order to visualize a nice interface for searching, browsing, and for the presentation of the data to the user. I developed my website using HTML, CSS, and Javascript, and I structured it into three files:

- *index.html*: the actual webpage displayed to the user
- *style.css*: the stylesheet used to add properties and characteristics to the elements displayed in the webpage
- *script.js*: the JavaScript file used to define the functions I needed in order to make requests to Solr and display the results to the user



```
alberma@MBP-di-Albe cse_website % tree
.
├── index.html
├── script.js
├── style.css
├── assets
│   ├── world.png
│   ├── favicon.ico
│   ├── wikipedia.png
│   ├── indeed.png
│   ├── usi_logo.png
│   ├── web_1.png
│   ├── logo.png
│   ├── scrapy.png
│   ├── britannica.png
│   ├── glass.png
│   ├── industry.png
│   ├── forbes.png
│   └── solr.png
alberma@MBP-di-Albe cse_website %
```

Figure 10: Website file tree

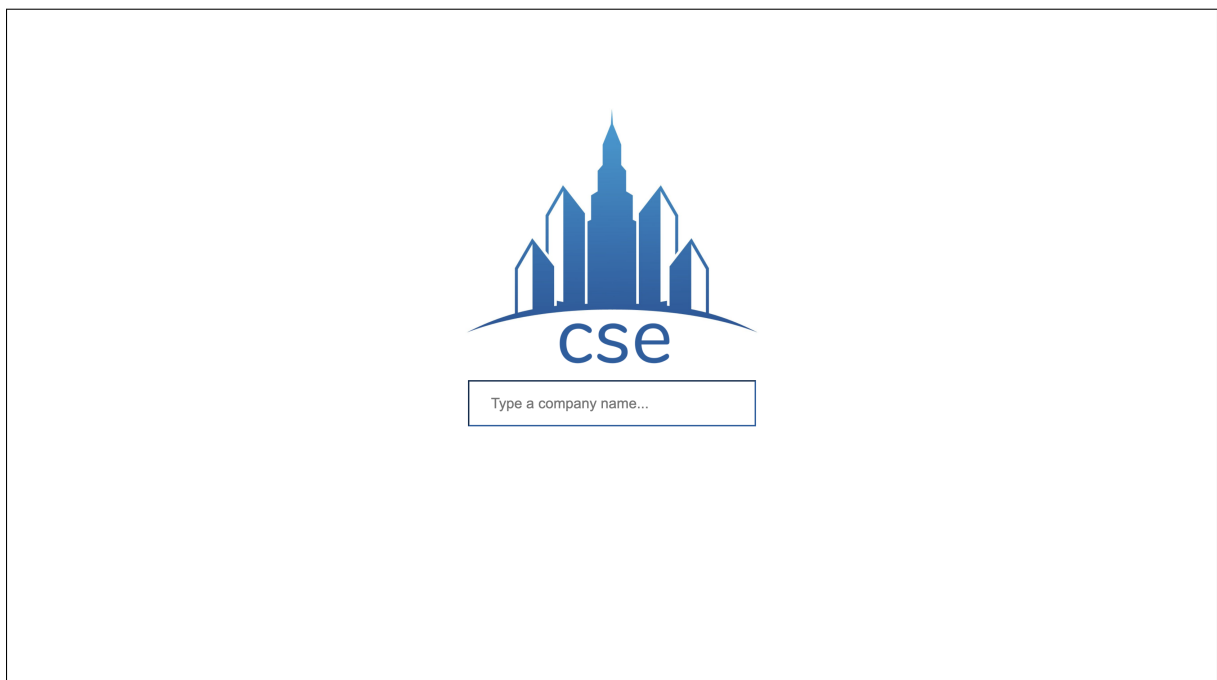



Figure 11: CSE index page

The results are displayed and presented in a single and dynamic page in a tabular format, as requested by the additional feature of the project, so that a user could look at different companies matching with his/her research with all the information concerning those companies. The companies are sorted by their rating in decreasing order.



Group

Companies	Rating	Country	Size	Industry
Swatch Group	3.7	Italy	N/A	Retail & Wholesale
Compass Group	3.2	France	N/A	Restaurants & Food Service
Calzedonia Group	3.3	Germany	N/A	Retail & Wholesale
Straumann Group	3.5	Austria	5,001 to 10,000	Manufacturing
Altria Group	N/A	United States	N/A	Tobacco
Hyundai Group	N/A	South Korea	N/A	Machine
Sumitomo Group	N/A	Japan	N/A	Mining
Mitsubishi Group	N/A	Japan	N/A	Military aircraft
Tata Group	N/A	India	N/A	Steel
Flick Group	N/A	Germany	N/A	Steel
Liggett Group Inc.	N/A	United States	N/A	Manufacturing
Continental Group, Inc.	N/A	United States	N/A	Insurance
ING Group NV	N/A	The Netherlands	N/A	Insurance

Figure 12: CSE results page

If there is not a match with the name typed by the user among the companies in the collection, the page displays a message to let the user know that his/her research did not retrieve any relevant results.

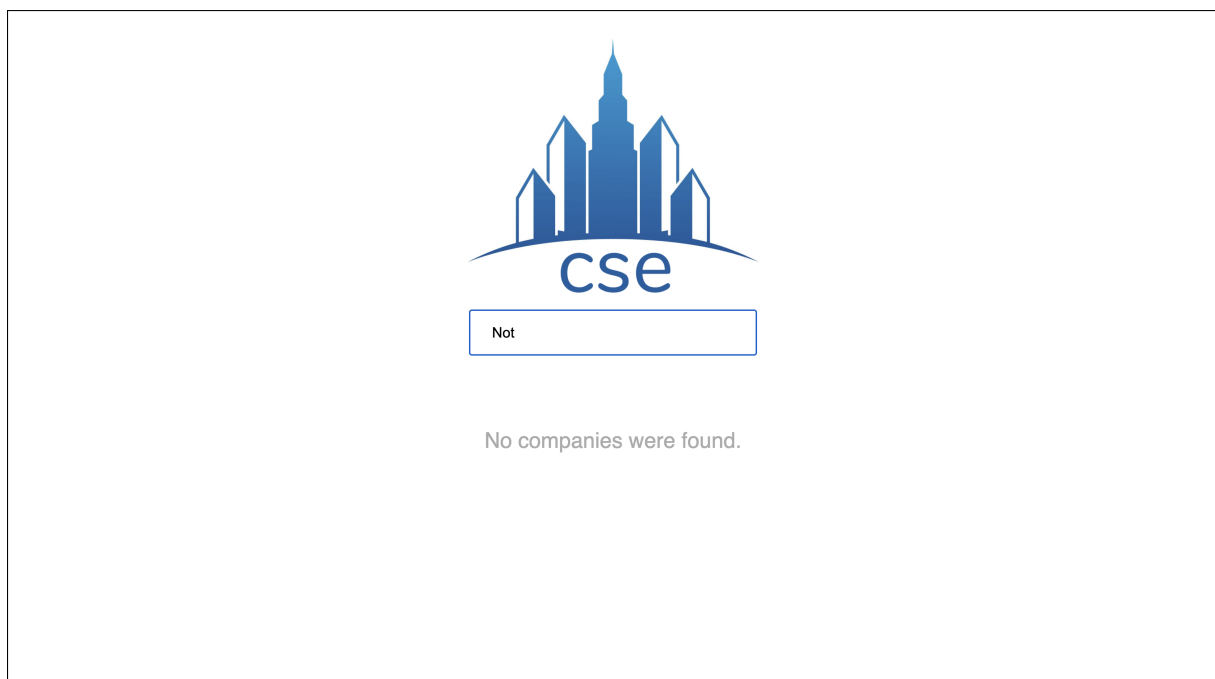


Figure 13: CSE results page

## 5. User Evaluation

For the user evaluation of my project, I asked three other students among those who are attending the Information Retrieval course this year. Since my project has not many features, I designed three tasks to test the following characteristics:

- Usability, meaning providing a condition to perform the tasks safely, effectively, and efficiently
- Results relevance, meaning the degree to which the results are related and useful to the user
- Interface quality

The tasks I designed are the following:

1. Type an arbitrary letter in the searchbox and choose three companies among the results, then check if the rating values were scraped correctly.
2. Type a white space, then select a company with a not available rating and a not available size and check if those values are not in the company webpage.
3. Analyze the differences between the data concerning the company “*Straumann Group*”, scraped from Indeed, and the company “*Groupon*”, scraped from Britannica.

I asked the users to choose a value from 0 to 5 depending on what they thought about each of the characteristics explained before. The results I obtained from the evaluation are displayed in the following tables:

	<b>Claudio Milanesi</b>
<b>Usability</b>	5, "simple and intuitive"
<b>Results Relevance</b>	5, "result accurate and precise"
<b>Interface Quality</b>	4.5, "straightforward interface"

	<b>Edoardo Riggio</b>
<b>Usability</b>	5, "easily usable"
<b>Results Relevance</b>	5, "not many companies but relevant"
<b>Interface Quality</b>	4.5, "nice interface"

	<b>Martino Giorgi</b>
<b>Usability</b>	3.5, "not many features"
<b>Results Relevance</b>	4, "sorting by rating is a good idea"
<b>Interface Quality</b>	4, "intuitive interface"

Finally, Claudio Milanesi advised me to *"add a message/popup to tell the user that the query must contain at least 1 character, or a white space"*.

## 6. Conclusion

In my opinion, the project was quite interesting, since I learnt how to use both Scrapy and Solr, which I never used before this course. Even if I couldn't manage to obtain a large amount of data as requested, I tried to gather as much information as possible concerning the companies I have in my collection and design a nice and intuitive GUI for displaying relevant results.