## Practicing Web scrapping

In the Web scraping tutorial, we learned how to use Python’s requests and BeautifulSoup libraries to make HTTP requests that allow data exchange from the web and to convert HTML web documents into Python data structures such as list, dictionaries and Pandas Dataframe, respectively. If you need a refresher, consult the workshop materials [here](https://tdmdal.github.io/Webscrapping/).

Now we can use these packages to download textual data from the web. One of the websites of increasing interest is VentureBeat, which qualifies as a leading media outlet to track buzzwords and number of mentions of companies and new ventures. According to their website, “VentureBeat is a leading source for transformative tech news and events that provide deep content to help business leaders make smart decisions and stay on top of breaking news.” This data can be used to aid the investment decision for the start-ups.

In order to perform our own analysis based on the content of these articles on VentureBeat, we will need to scrape this data first. Here is what is required to scrape this data from their website:

**Part A**

1. Perform HTTP request to VentureBeat’s website using Python’s requests package. (use .get method)
2. Convert the html content retrieved from step 1 into a BeautifulSoup object. (use BeautifulSoup function with html.parser to parse HTML)
3. Find the HTML tags and attribute that contains the url link to each of the article on the main web page, extract them and store these links in a list. (Use .find\_all method)

**Part B**

Now that we have the links to each of the articles itself, we can extract the text contents of these articles

1. Perform HTTP request to each of these articles. (Use list comprehension or for loop and .get method)
2. Convert each html output from step 1 into BeautifulSoup objects. (Use list comprehension or for loop and BeautifulSoup function with html.parser to parse HTML)
3. Find the HTML tag(s) and attribute that contains the text of the article, extract them and store it into a list of lists. (Use .find\_all method)

**Part C**

Now we should have a **nested list** – text of each articles is inside a list and these lists are inside another giant list.

Ex.: [ [text of article 1], [text of article 2], [text of article 3], … ]

1. Take the list of links from Part A and the nested list from Part B and convert it into a pandas dataframe. Hence, we will have two columns - ‘url’ and ‘text body’.

If you have any questions, please email me at pythonhelp@rotman.utoronto.ca or drop-by my office at Rotman 3001.