**Introduction:**

In this project, we are given 50 HTML files of SEC forms, and the objective is to extract EPS data from each file. The inherent challenge with this task is that each file has a different variation of labels for EPS, e.g., "Net earnings per share – basic," "Earnings (loss) per common share," etc. To identify a label for each file, we resort to a machine learning approach. Instead of applying NLP to the text descriptions in the files, we use cosine similarity for the table data entries. The benefit of this approach is that earnings per share is not necessarily described in the text portion of the file but is guaranteed to be included in the table data, as per GAAP regulations. After extracting the cell containing the EPS label, we need to find the actual EPS numbers in the adjacent cells. This raises another challenge, which is to build a robust methodology to identify and extract EPS numbers after locating the row containing the EPS label in the tables.

**Acquiring label from parsed table:**

First, we read through the folder containing all 50 HTML files and store the contents and filenames in two lists. Next, we manually build a dictionary using EPS labels from randomly chosen files. We then create a function that parses the table data from each HTML using the Beautiful Soup package and returns the label cell most closely associated with our dictionary in terms of cosine similarity score. Using the function ‘find\_most\_similar\_cell’ and the labels it outputs for each HTML, we fine-tune our dictionary to optimize accuracy. We ensure the label output contains at least three words, as 'Earnings per Share' has a length of three, to make the label selection more robust and avoid shorter labels such as 'per share' or 'basic'.

**Acquire EPS using labeled cell:**

Once we have identified the label cell, the cell containing the EPS data should be within the same row or, at most, one row below it in the table. To address this, we developed the find\_value\_after\_label function to scan these two rows for the target value. First, we examine the initial row to locate the first cell containing numerical data, using the parse\_number function to remove dollar sign and parentheses, and extract a positive or negative number of EPS. If no numerical cells are found in the first row, we then proceed to the second row to extract numbers. Should this algorithm fail to locate any numbers, it returns 0.

**Outlier detection:**

After processing all 50 HTML files, we review our output table to identify any potential issues. We notice that the machine learning model sometimes outputs label cells such as 'Per common share', which can lead to erroneously high EPS values in the thousands. To address this, we check the absolute values in the EPS column and perform a quality check on our results. If a value appears too extreme, we replace it with 0.