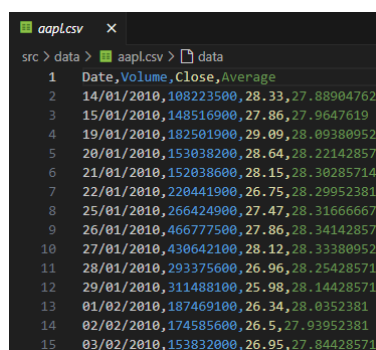


## 1 Introduction:

In the dynamic landscape of financial markets, the rising complexity of data demands effective tools for analysis. Data visualization, transcending traditional numerical reports, has become indispensable for swiftly interpreting and acting on information. This project highlights the crucial role of visualization in revealing actionable insights, underscoring its importance in modern financial analysis.

## 2 Data:

Data used for this project represents stock values between 2010 and 2014. The Database contains 4 columns named as Date, Volume, Close, Average



	Date	Volume	Close	Average
1	14/01/2010	108223500	28.33	27.88904762
2	15/01/2010	148516900	27.86	27.9647619
3	19/01/2010	182501900	29.09	28.09380952
4	20/01/2010	153038200	28.64	28.22142857
5	21/01/2010	152038600	28.15	28.30285714
6	22/01/2010	220441900	26.75	28.29952381
7	25/01/2010	266424900	27.47	28.31666667
8	26/01/2010	466777500	27.86	28.34142857
9	27/01/2010	430642100	28.12	28.33300952
10	28/01/2010	293375600	26.96	28.25428571
11	29/01/2010	311488100	25.98	28.14428571
12	01/02/2010	187469100	26.34	28.0352381
13	02/02/2010	174585600	26.5	27.93952381
14	03/02/2010	153832000	26.95	27.84428571

- Date: serves as a chronological anchor, providing a timeline for the stock market data.
- Volume: quantifies the number of shares traded on a given date.
- Close: signifies the closing price of a stock on a particular date.
- Average: represents the average stock value over a specified period, offering a smoothed-out view of price movements.

## 3 Visualisation of Stocks:

To visualize the dataset, I started by parsing the column's values using the function **type()** defined in the Js code.

```
function type(d) {  
  return {  
    date : parseDate(d.Date),  
    price : +d.Close,  
    average : +d.Average,  
    volume : +d.Volume,  
  }  
}
```

Then I worked on a function, that was called **setupScalesAndAxes()**, that configured time and linear scales along with corresponding axes for the visualization.

```
function setupScalesAndAxes() {  
  x = d3.time.scale().range([0, width]);  
  x2 = d3.time.scale().range([0, width]);  
  y = d3.scale.linear().range([height, 0]);  
  y1 = d3.scale.linear().range([height, 0]);  
  y2 = d3.scale.linear().range([height2, 0]);  
}
```

```

y3 = d3.scale.linear().range([60, 0]);

xAxis = d3.svg.axis().scale(x).orient('bottom');
xAxis2 = d3.svg.axis().scale(x2).orient('bottom');
yAxis = d3.svg.axis().scale(y).orient('left');

};

```

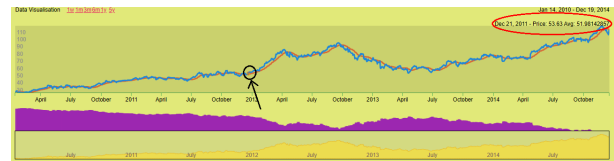
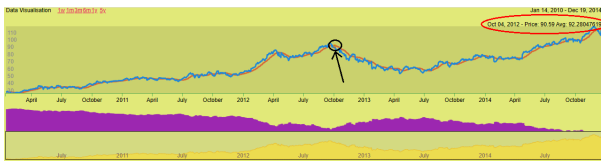
Then I worked on a function, that was called **createLineAndAreaGenerators()**, to initialize D3 line and area generators for representing stock prices and averaging over time.

After that, I started working more on the visualization. I started creating SVG elements, defining axes and handling user interactions.

```

function mousemove() {
    var x0 = x.invert(d3.mouse(this)[0]);
    var i = bisectDate(data, x0, 1);
    var d0 = data[i - 1];
    var d1 = data[i];
    var d = x0 - d0.date > d1.date - x0 ? d1 : d0;
    helperText.text(legendFormat(new Date(d.date)) + ' - Price: ' + d.price + ' Avg: ' + d.average);
    priceTooltip.attr('transform', 'translate(' + x(d.date) + ',' + y(d.price) + ')');
    averageTooltip.attr('transform', 'translate(' + x(d.date) + ',' + y(d.average) + ')');
}

```



Another quite interesting functionality is the function that was called **brushed()**. This function enables selecting a specific time range in the visualization. This feature makes the work more interactive as selecting a certain time range will update the chart created showing stock values for that specific period.

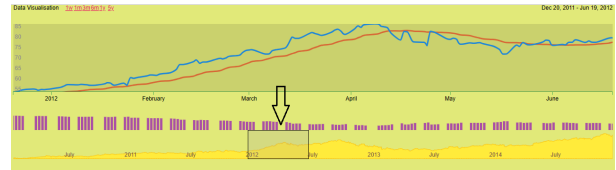
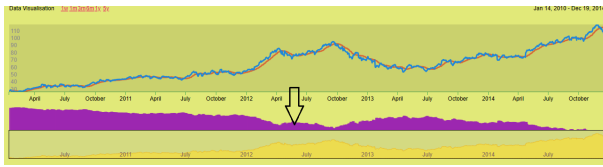
```

function brushed() {
    var ext = brush.extent();
    if (!brush.empty()) {
        x.domain(brush.empty() ? x2.domain() : brush.extent());
        y.domain([
            d3.min(data.map(function(d) { return (d.date >= ext[0] && d.date <= ext[1]) ? d.price : max; })),
            d3.max(data.map(function(d) { return (d.date >= ext[0] && d.date <= ext[1]) ? d.price : min; }))
        ]);
        range.text(legendFormat(new Date(ext[0])) + ' - ' + legendFormat(new Date(ext[1])));
        focusGraph.attr('x', function(d, i) { return x(d.date); });

        var days = Math.ceil((ext[1] - ext[0]) / (24 * 3600 * 1000))
        focusGraph.attr('width', (40 > days) ? (40 - days) * 5 / 6 : 5)
    }

    priceChart.attr('d', priceLine);
    averageChart.attr('d', avgLine);
    focus.select('.x.axis').call(xAxis);
    focus.select('.y.axis').call(yAxis);
}

```



## 4 The Chatbot:

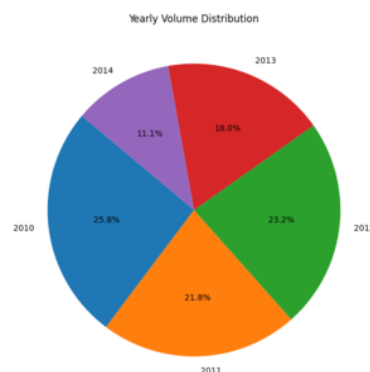
Then, I incorporated a simple chatbot feature that engages users in a conversation about stock market-related queries. The chatbot interface is created dynamically and includes a dropdown menu with predefined questions about stock market concepts. Users can select a question from the dropdown, click the "Demander" (Ask) button, and the chatbot responds with relevant information.

User: How do I read and interpret a stock chart?

A stock chart visualizes price movements over time. The x-axis represents time, the y-axis shows prices, and candlesticks or bars depict price changes. Look for trends, support/resistance levels, and consider indicators like volume and technical tools for insights into market movements.

## 5 Pie Chart:

In this part, I visualized the percentage of number of stocks in each year in a pie chart. Such a presentation is essential to understand the evolution of the stock market.



## 6 Other elements that could be added:

One of the features that can be added to the project is real-time data visualization. By utilizing a specific API key, we can retrieve up-to-the-minute stock values. This addition is particularly intriguing, as it opens the door to incorporating forecasting features into the visualization.

Another feature could involve gamifying the visualization experience. We can simulate the role of a trader, allowing users to engage in virtual buying and selling of stocks. This interactive approach provides users with the opportunity to explore stock trading in an imaginative and simulated manner.

## 7 Conclusion:

This project highlights the importance of data visualization in financial analysis, offering a user-friendly representation of stock values. Interactive features like brushing for specific time ranges and a dynamic chatbot enhance the overall experience. The addition of a pie chart provides a concise overview of stock distribution. Future improvements, like real-time data visualization and gamification, could further enhance the project's functionality for advanced analysis and engagement.