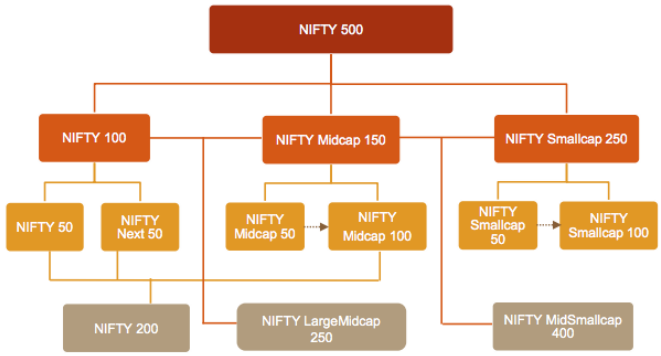
What is NIFTY 50 ?

The NIFTY 50 index is National Stock Exchange of India's benchmark broad based stock market index for the Indian equity market. NIFTY 50 stands for National Index Fifty, and represents the weighted average of 50 Indian company stocks in 17 sectors. It is one of the two main stock indices used in India, the other being the BSE Sensex. Source:Wikipedia



Dataset

The dataset consists of 13 files, here is a brief overview of each file:

INDIAVIX

India VIX is a volatility index based on the NIFTY Index Option prices.Volatility Index is a measure of market's expectation of volatility over the near term. Volatility is often described as the "rate and magnitude of changes in prices" and in finance often referred to as risk. Volatility Index is a measure, of the amount by which an underlying Index is expected to fluctuate, in the near term,Source

NIFTY 50, NIFTY 100 and NIFTY 500

- NIFTY 500 - It represents the top 500 companies based on full market capitalisation from the eligible universe
- NIFTY 100 - This represents the top 100 companies (i.e. from 1 to 100) from within the NIFTY 500. This index basically tries to track the performance of companies having large market caps.
- NIFTY 50 - This represents the first 50 companies from the NIFTY 100.

NIFTY SMALL CAP & MID CAP

- NIFTY SMALLCAP - This index measures the performance of small-cap companies.
- NIFTY MIDCAP - This index tries to measure the performance of mid-cap companies.

NIFTY NEXT 50

This includes the remaining 50 companies from NIFTY 100 after excluding the NIFTY 50 companies. These are also called as NIFTY Junior.

NIFTY SECTORAL INDICES

This includes NIFTY AUTO,NIFTY BANK, NIFTY FMCG, NIFTY IT,NIFTY METAL, NIFTY PHARMA These Indices are designed to reflect the behavior and performance of the segment that they reflect i.e automobiles, bank, pharma etc.

Importing the required libraries

In [1]:
```
import numpy as np
import pandas as pd
import plotly.express as px
import chart_studio.plotly as py
import plotly.graph_objs as go
from plotly.offline import iplot
import cufflinks
cufflinks.go_offline()
cufflinks.set_config_file(world_readable=True, theme='pearl')
from datetime import datetime
```

Loading the NIFTY50 dataset

In [2]:
```
nifty_50 = pd.read_csv('NIFTY 50.csv', parse_dates = ["Date"])
nifty_50.head()
```

Out[2]:

	Date	Open	High	Low	Close	Volume	Turnover	P/E	P/B	Div Yield
0	2000-01-03	1482.15	1592.90	1482.15	1592.2	25358322.0	8.841500e+09	25.91	4.63	0.95
1	2000-01-04	1594.40	1641.95	1594.40	1638.7	38787872.0	1.973690e+10	26.67	4.76	0.92
2	2000-01-05	1634.55	1635.50	1555.05	1595.8	62153431.0	3.084790e+10	25.97	4.64	0.95
3	2000-01-06	1595.80	1639.00	1595.80	1617.6	51272875.0	2.531180e+10	26.32	4.70	0.94
4	2000-01-07	1616.60	1628.25	1597.20	1613.3	54315945.0	1.914630e+10	26.25	4.69	0.94

In [3]:
```
nifty_50.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5061 entries, 0 to 5060
Data columns (total 10 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Date       5061 non-null   datetime64[ns]
 1   Open       5061 non-null   float64
 2   High       5061 non-null   float64
 3   Low        5061 non-null   float64
 4   Close      5061 non-null   float64
 5   Volume     5060 non-null   float64
 6   Turnover   5060 non-null   float64
 7   P/E        5061 non-null   float64
 8   P/B        5061 non-null   float64
 9   Div Yield  5061 non-null   float64
dtypes: datetime64[ns](1), float64(9)
memory usage: 395.5 KB
```

About the Stock Data

Now that our data has been converted into the desired format, let's take a look at its various columns for further analysis.

- **The Open and Close columns** indicate the opening and closing price of the stocks on a particular day.
- **The High and Low columns** provide the highest and the lowest price for the stock on a particular day, respectively.
- **The Volume column** tells us the total volume of stocks traded on a particular day.
- **The Turnover column** refers to the total value of stocks traded during a specific period of time. The time period may be annually, quarterly, monthly or daily
- **P/E** also called as the price-earnings ratio relates a company's share price to its earnings per share.
- **P/B** also called as Price-To-Book ratio measures the market's valuation of a company relative to its book value.
- **Div Yield** or the dividend yield is the amount of money a company pays shareholders (over the course of a year) for owning a share of its stock divided by its current stock price—displayed as a percentage.

Missing values

In [4]:
```
nifty_50.isnull().sum()
```

Out[4]:
```
Date         0
Open         0
High         0
Low          0
Close        0
Volume       1
Turnover     1
P/E          0
P/B          0
Div Yield    0
dtype: int64
```

In [5]:
```
nifty_50.fillna(method = 'ffill', inplace = True)
```

method='ffill': This parameter specifies the method for filling missing values.

In this case, 'ffill' stands for "forward fill.

It means that missing values will be filled using the last known (non-missing) value in the column.

In [6]:
```
nifty_50.isnull().sum()
```

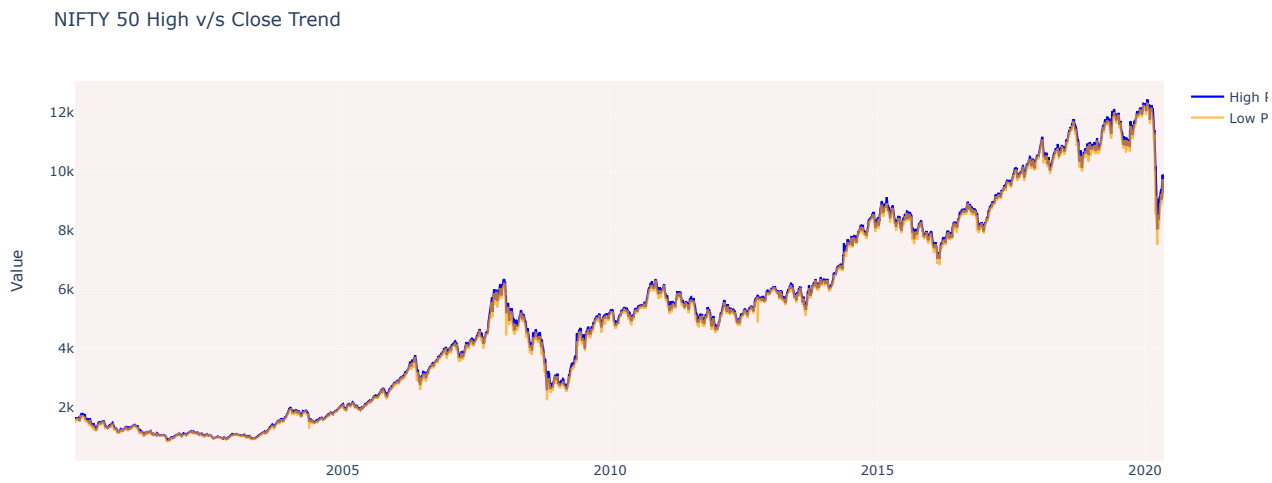
Out[6]:
```
Date         0
Open         0
High         0
Low          0
Close        0
Volume       0
Turnover     0
P/E          0
P/B          0
Div Yield    0
dtype: int64
```

Visualising the NIFTY50 data

In [7]:
```
fig = go.Figure()
fig.add_trace(go.Scatter(x = nifty_50['Date'], y = nifty_50['High'], name = 'High Price',
    line = dict(color = 'blue')))

fig.add_trace(go.Scatter(x = nifty_50['Date'], y = nifty_50['Low'], name = 'Low Price',
    line = dict(color = 'orange'), opacity = 0.7))

fig.update_layout(title_text='NIFTY 50 High v/s Close Trend',plot_bgcolor = 'rgb(250, 242, 242)', yaxis_title = 'Value')
fig.show()
```



In [8]:
```
fig = go.Figure()
fig.add_trace(go.Scatter(x = nifty_50['Date'], y = nifty_50['Close'], name = 'Closing Price',
    line=dict(color='green'), opacity=0.8))

fig.update_layout(title_text='NIFTY 50 Closing Price Trend', plot_bgcolor = 'rgb(250, 242, 242)', yaxis_title = 'Value')
fig.show()
```

P/E vs P/B Ratio : Which one to use?

P/E ratio is a popular measure of how expensive a company's stock is. The Price-to-Earnings (P/E) ratio measures a company's current stock price relative to its earnings per share (EPS). It helps investors assess how much they are willing to pay for each dollar of earnings generated by the company. A higher P/E ratio may indicate higher growth expectations or overvaluation, while a lower ratio may suggest undervaluation or lower growth prospects.

$$\text{P/E Ratio} = \frac{\text{Market value per share}}{\text{Earnings per share}}$$

The **P/B ratio** (P/B) ratio compares a company's stock price to its book value per share, indicating how the market values the company relative to its net assets. A lower P/B ratio suggests the stock may be undervalued, while a higher ratio may imply overvaluation.P/B ratio is used by value investors to identify potential investments and P/B ratios under 1 are typically considered solid investments.

$$P/B\ Ratio = \frac{Market\ Price\ per\ Share}{Book\ Value\ per\ Share}$$

Whether the Price-to-Earnings (P/E) ratio or Price-to-Book (P/B) ratio is better depends on the industry's characteristics and the company's financial structure. P/E ratio reflects earnings and growth expectations, making it useful for assessing growth-oriented industries. P/B ratio, comparing market value to book value, is valuable for asset-heavy industries like banking. Considering both ratios together provides a more comprehensive view of a company's financial health, growth potential, and market perception, leading to more informed investment decisions.

In [9]:
```
fig = go.Figure()
fig.add_trace(go.Scatter(x = nifty_50['Date'], y = nifty_50['P/E'], name = 'P/E Ratio',
    line = dict(color ='green')))

fig.add_trace(go.Scatter(x = nifty_50['Date'], y = nifty_50['P/B'], name = 'P/B Ratio',
    line = dict(color ='orange')))

fig.update_layout(title_text = 'P/E vs P/B Ratio', plot_bgcolor = 'rgb(250, 242, 242)', yaxis_title = 'Value')
fig.show()
```

Market performance 2014 onwards

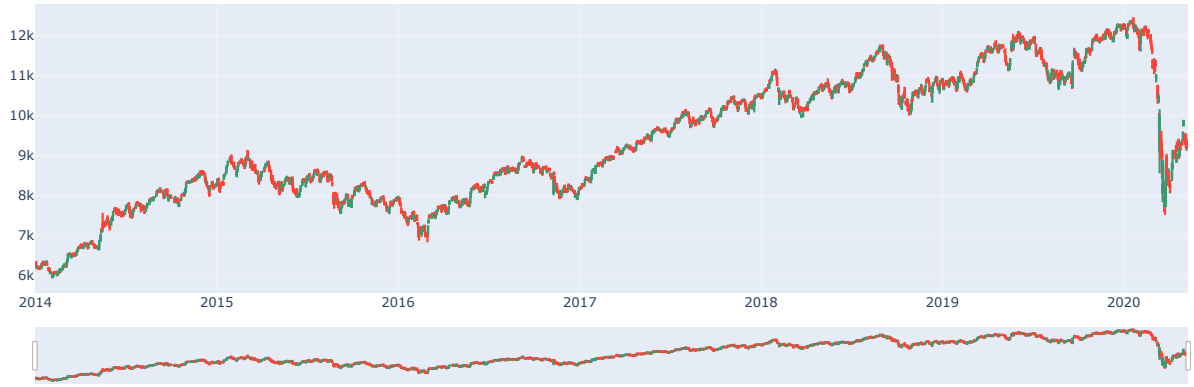
In [10]:
```
nifty_50_2014 = nifty_50[nifty_50['Date'] >= '2014-01-01']
nifty_50_2014.head()
```

Out[10]:

	Date	Open	High	Low	Close	Volume	Turnover	P/E	P/B	Div Yield
3496	2014-01-01	6323.80	6327.2	6298.25	6301.65	69567668.0	2.015360e+10	18.69	2.99	1.49
3497	2014-01-02	6301.25	6358.3	6211.30	6221.15	158132556.0	5.249790e+10	18.46	2.95	1.50
3498	2014-01-03	6194.55	6221.7	6171.25	6211.15	139043889.0	5.369420e+10	18.43	2.95	1.51
3499	2014-01-06	6220.85	6224.7	6170.25	6191.45	118344976.0	4.722670e+10	18.37	2.94	1.51
3500	2014-01-07	6203.90	6221.5	6144.75	6162.25	138559000.0	5.696600e+10	18.28	2.92	1.52

In [11]:
```
df = nifty_50_2014
fig = go.Figure(data = [go.Candlestick(x=df['Date'], open = df['Open'], high = df['High'],
                low = df['Low'], close = df['Close'])])

fig.show()
```



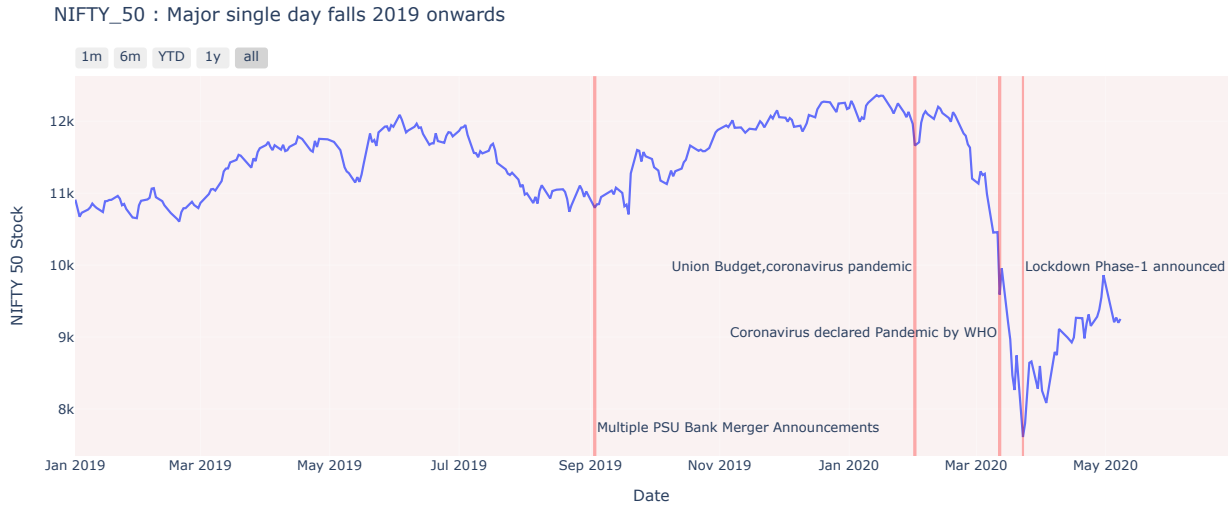
NIFTY_50 : Major single day falls (2019 onwards)

In [12]:
```
nifty_50_2019 = nifty_50[nifty_50['Date'] >= '2019-01-01']
```

In [13]:
```
fig = px.line(nifty_50_2019, x='Date', y='Close', title='Time Series with Range Slider and Selectors')

fig.update_xaxes(
    rangeslider_visible=False,
    rangeselector=dict(
        buttons=list([
            dict(count=1, label="1m", step="month", stepmode="backward"),
            dict(count=6, label="6m", step="month", stepmode="backward"),
            dict(count=1, label="YTD", step="year", stepmode="todate"),
            dict(count=1, label="1y", step="year", stepmode="backward"),
            dict(step="all")
        ])
    )
)

fig.update_layout(plot_bgcolor='rgb(250, 242, 242)',
    title='NIFTY_50 : Major single day falls 2019 onwards',
    yaxis_title='NIFTY 50 Stock',
    shapes = [dict(x0='2020-03-23', x1='2020-03-23', y0=0, y1=1, xref='x', yref='paper', line_width=2,opacity=0.3,line_color='red',editable=False),
              dict(x0='2019-09-3', x1='2019-09-3', y0=0, y1=1, xref='x', yref='paper',line_width=3,opacity=0.3,line_color='red'),
              dict(x0='2020-02-1', x1='2020-02-1', y0=0, y1=1, xref='x', yref='paper',line_width=3,opacity=0.3,line_color='red'),
              dict(x0='2020-03-12', x1='2020-03-12', y0=0, y1=1, xref='x', yref='paper',line_width=3,opacity=0.3,line_color='red')],
    annotations=[dict(x='2020-03-23', y=0.5, xref='x', yref='paper',
                      showarrow=False, xanchor='left', text='Lockdown Phase-1 announced'),
                 dict(x='2019-09-3', y=0.05, xref='x', yref='paper',
                      showarrow=False, xanchor='left', text='Multiple PSU Bank Merger Announcements'),
                 dict(x='2020-02-1', y=0.5, xref='x', yref='paper',
                      showarrow=False, xanchor='right', text='Union Budget,coronavirus pandemic'),
                 dict(x='2020-03-12', y=0.3, xref='x', yref='paper',
                      showarrow=False, xanchor='right', text='Coronavirus declared Pandemic by WHO')]
)
fig.show()
```



The NIFTY 50 index experienced a significant decline in the year 2020, prominently reflected in the graph above. The most notable impact occurred during the commencement of the initial phase of the lockdown.

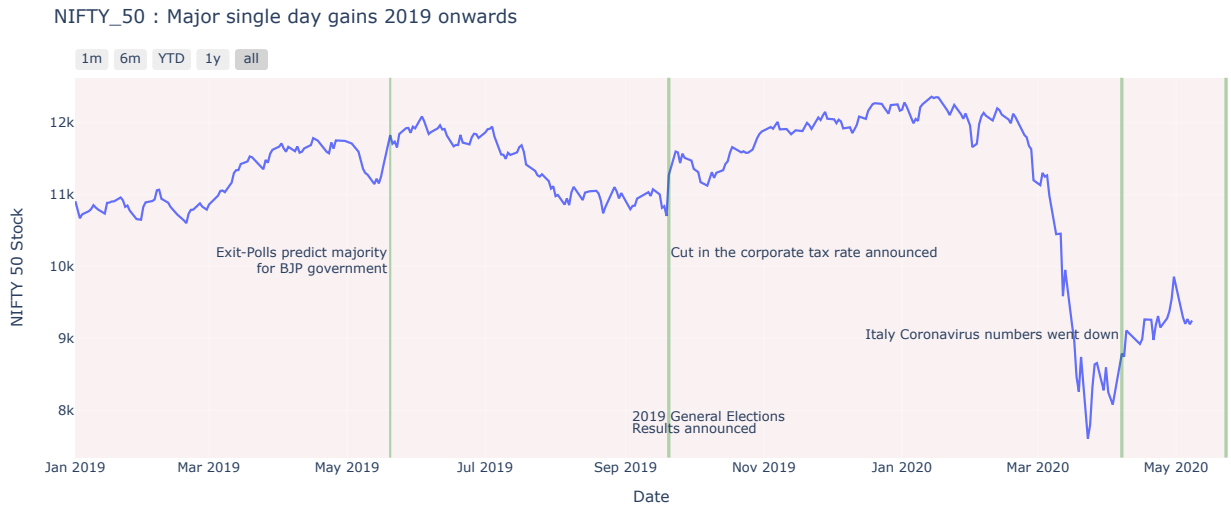
NIFTY50 : Major single day gains (2019 onwards)

In [14]:
```
fig = px.line(nifty_50_2019, x='Date', y='Close', title='Time Series with Range Slider and Selectors')

fig.update_xaxes(
    rangeslider_visible=False,
    rangeselector=dict(
        buttons=list([
            dict(count=1, label="1m", step="month", stepmode="backward"),
            dict(count=6, label="6m", step="month", stepmode="backward"),
            dict(count=1, label="YTD", step="year", stepmode="todate"),
            dict(count=1, label="1y", step="year", stepmode="backward"),
            dict(step="all")
        ])
    )
)

fig.update_layout(plot_bgcolor='rgb(250, 242, 242)',
    title='NIFTY_50 : Major single day gains 2019 onwards',
    yaxis_title='NIFTY 50 Stock',
    shapes = [dict(x0='2019-05-20', x1='2019-05-20', y0=0, y1=1, xref='x', yref='paper', line_width=2,opacity=0.3,line_color='green',editable=False),
              dict(x0='2020-05-23', x1='2020-05-23', y0=0, y1=1, xref='x', yref='paper',line_width=3,opacity=0.3,line_color='green'),
              dict(x0='2019-09-20', x1='2019-09-20', y0=0, y1=1, xref='x', yref='paper',line_width=3,opacity=0.3,line_color='green'),
              dict(x0='2020-04-07', x1='2020-04-07', y0=0, y1=1, xref='x', yref='paper',line_width=3,opacity=0.3,line_color='green')],
    annotations=[dict(x='2019-05-20', y=0.54, xref='x', yref='paper',
                      showarrow=False, xanchor='right', text='Exit-Polls predict majority'),
                 dict(x='2019-05-20', y=0.5, xref='x', yref='paper',
                      showarrow=False, xanchor='right', text='for BJP government'),
                 dict(x='2019-09-3', y=0.08, xref='x', yref='paper',
                      showarrow=False, xanchor='left', text='2019 General Elections'),
                 dict(x='2019-09-3', y=0.05, xref='x', yref='paper',
                      showarrow=False, xanchor='left', text='Results announced'),
                 dict(x='2019-09-20', y=0.54, xref='x', yref='paper',
```

```
                showarrow=False, xanchor='left', text='Cut in the corporate tax rate announced'),
            dict(x='2020-04-07', y=0.3, xref='x', yref='paper',
                showarrow=False, xanchor='right', text='Italy Coronavirus numbers went down')]
)
fig.show()
```



Performance of other NIFTY Sectoral Indices in 2020

In [15]:
```
nifty_auto = pd.read_csv('NIFTY AUTO.csv',parse_dates=["Date"])
nifty_bank = pd.read_csv('NIFTY BANK.csv',parse_dates=["Date"])
nifty_fmcg = pd.read_csv('NIFTY FMCG.csv',parse_dates=["Date"])
nifty_IT = pd.read_csv('NIFTY IT.csv',parse_dates=["Date"])
nifty_metal = pd.read_csv('NIFTY METAL.csv',parse_dates=["Date"])
nifty_pharma = pd.read_csv('NIFTY PHARMA.csv',parse_dates=["Date"])
```

In [16]:
```
# filling in missing values using forward fill method

nifty_auto.fillna(method='ffill',inplace=True)
nifty_bank.fillna(method='ffill',inplace=True)
nifty_fmcg.fillna(method='ffill',inplace=True)
nifty_IT.fillna(method='ffill',inplace=True)
nifty_metal.fillna(method='ffill',inplace=True)
nifty_pharma.fillna(method='ffill',inplace=True)
```

- NIFTY Auto Index:

The Nifty Auto Index tracks the performance of the Automobiles sector, encompassing manufacturers of cars, motorcycles, heavy vehicles, auto ancillaries, and tyres, among others.

- NIFTY Bank Index:

Nifty Bank Index includes highly liquid and large-cap Indian Banking stocks, serving as a benchmark that reflects the capital market performance of Indian Banks for investors and market intermediaries.

- NIFTY FMCG Index:

The Nifty FMCG Index comprises up to 15 companies that manufacture Fast Moving Consumer Goods (FMCG) products.

- NIFTY IT Index:

Companies in the Nifty IT Index derive over 50% of their turnover from IT-related activities such as IT infrastructure, IT education and software training, telecommunication services, networking infrastructure, software development, hardware manufacturing, vending, support, and maintenance.

- NIFTY Metal Index:

The Nifty Metal Index represents the behavior and performance of the Metals sector, including mining. It consists of up to 15 stocks listed on the National Stock Exchange.

- NIFTY Pharma Index:

Nifty Pharma Index is designed to measure the performance of Pharmaceutical companies operating in this sector.

Comparing Closing prices of different sectoral indices

In [17]:
```
nifty_auto_2019 = nifty_auto[nifty_auto['Date'] > '2019-12-31']
nifty_bank_2019 = nifty_bank[nifty_bank['Date'] > '2019-12-31']
nifty_fmcg_2019 = nifty_fmcg[nifty_fmcg['Date'] > '2019-12-31']
nifty_IT_2019 = nifty_IT[nifty_IT['Date'] > '2019-12-31']
nifty_metal_2019 = nifty_metal[nifty_metal['Date'] > '2019-12-31']
nifty_pharma_2019 = nifty_pharma[nifty_pharma['Date'] > '2019-12-31']

d = {'NIFTY Auto index': nifty_auto_2019['Close'].values,
     'NIFTY Bank index': nifty_bank_2019['Close'].values,
     'NIFTY FMCG index': nifty_fmcg_2019['Close'].values,
     'NIFTY IT index': nifty_IT_2019['Close'].values,
     'NIFTY Metal index': nifty_metal_2019['Close'].values,
     'NIFTY Pharma index': nifty_pharma_2019['Close'].values,
    }
```

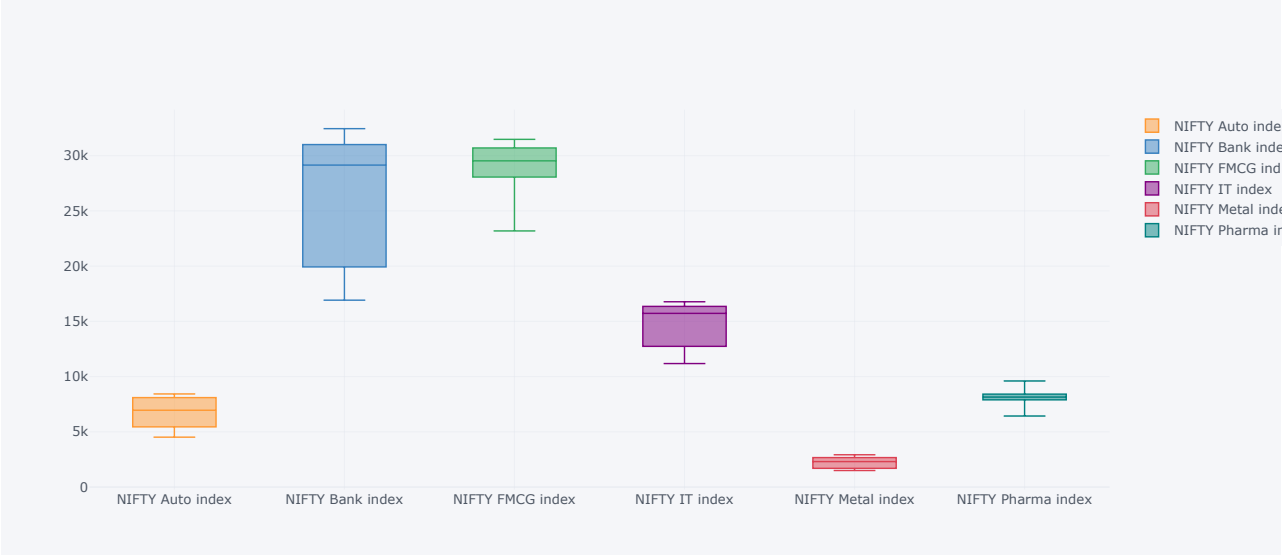
In [18]:
```
df = pd.DataFrame(data = d)
df.index = nifty_auto_2019['Date']
```

```
df.head()
```

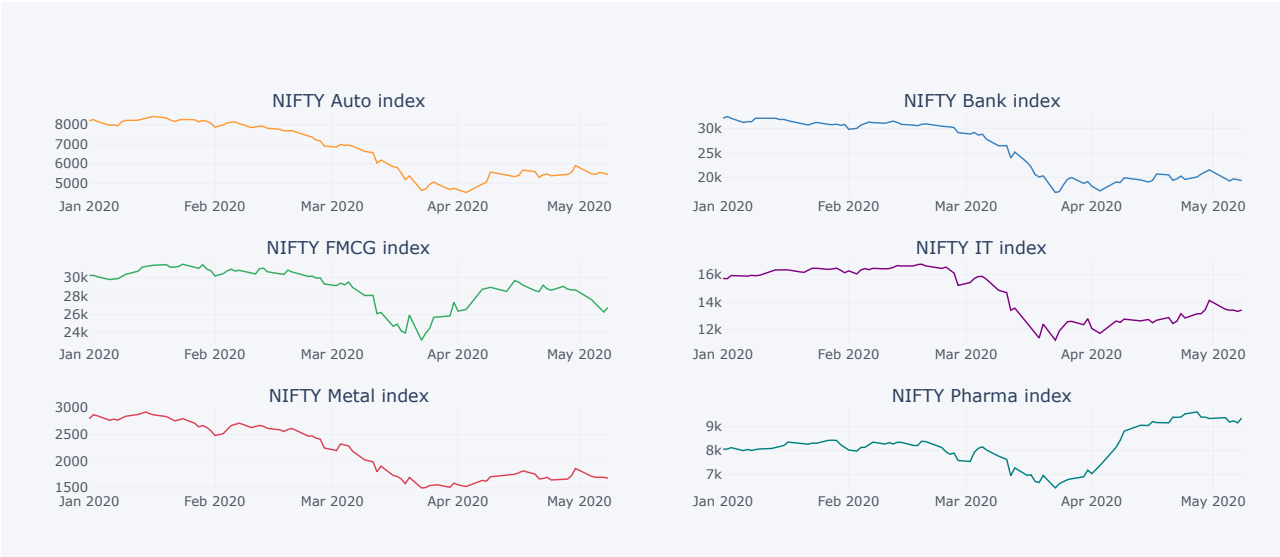
Out[18]:

Date	NIFTY Auto index	NIFTY Bank index	NIFTY FMCG index	NIFTY IT index	NIFTY Metal index	NIFTY Pharma index
2020-01-01	8210.10	32102.90	30234.25	15722.15	2796.05	8047.10
2020-01-02	8267.45	32443.85	30266.20	15709.65	2869.90	8053.95
2020-01-03	8168.15	32069.25	30109.25	15936.60	2848.35	8111.95
2020-01-06	7978.75	31237.15	29799.30	15879.80	2765.75	7987.35
2020-01-07	8002.50	31399.40	29861.80	15895.20	2785.90	8036.50

In [19]:
```
df.iplot(kind = 'box')
```



In [20]:
```
fig = df.iplot(asFigure = True, subplots = True, subplot_titles = True, legend = False)
fig.show()
```

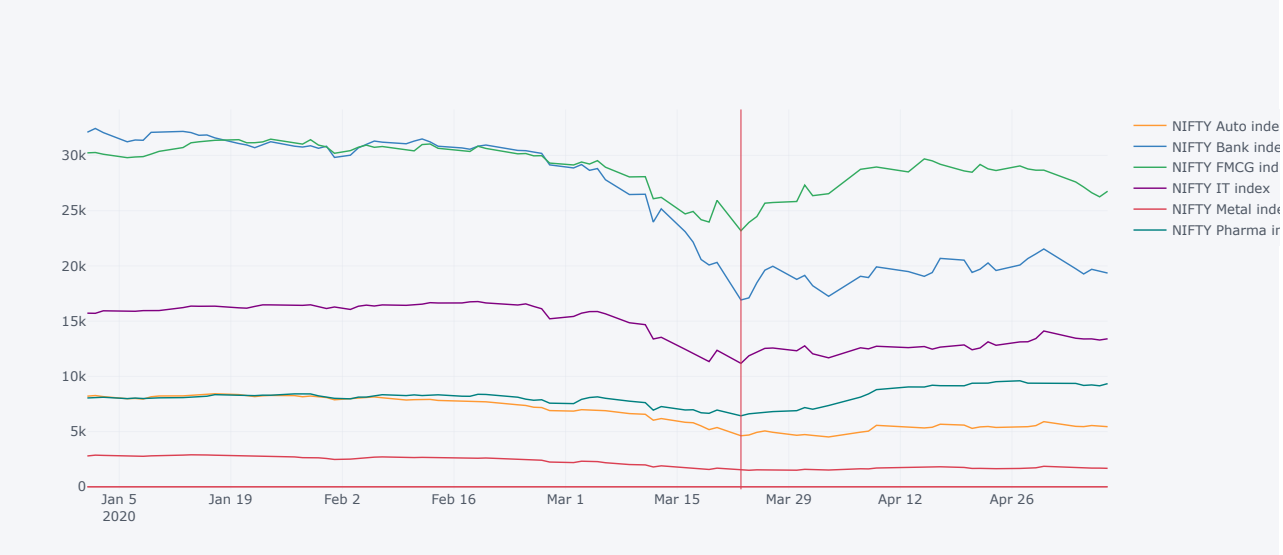


During 2020, all sectoral indices experienced an impact, as evident from the above data. However, it is notable that the FMCG (Fast Moving Consumer Goods) and Pharma sectors have shown signs of recovery or rebounding. Despite the challenging economic conditions, these sectors have displayed resilience and a potential return to growth, reflecting their essential nature and consumer demand during uncertain times.

COVID-19 pandemic Lockdown Effect

On 24th March 2020, in response to the COVID-19 pandemic, the Government of India, led by Prime Minister Narendra Modi, implemented a comprehensive nationwide lockdown for a duration of 21 days. This extraordinary measure aimed to curb the spread of the virus and protect the health of India's entire population, which amounts to approximately 1.3 billion people. The lockdown entailed significant restrictions on movement, public gatherings, and non-essential activities across the country. The decision was taken as a precautionary and preventive strategy to mitigate the impact of the pandemic and ensure public safety and well-being.[Source]

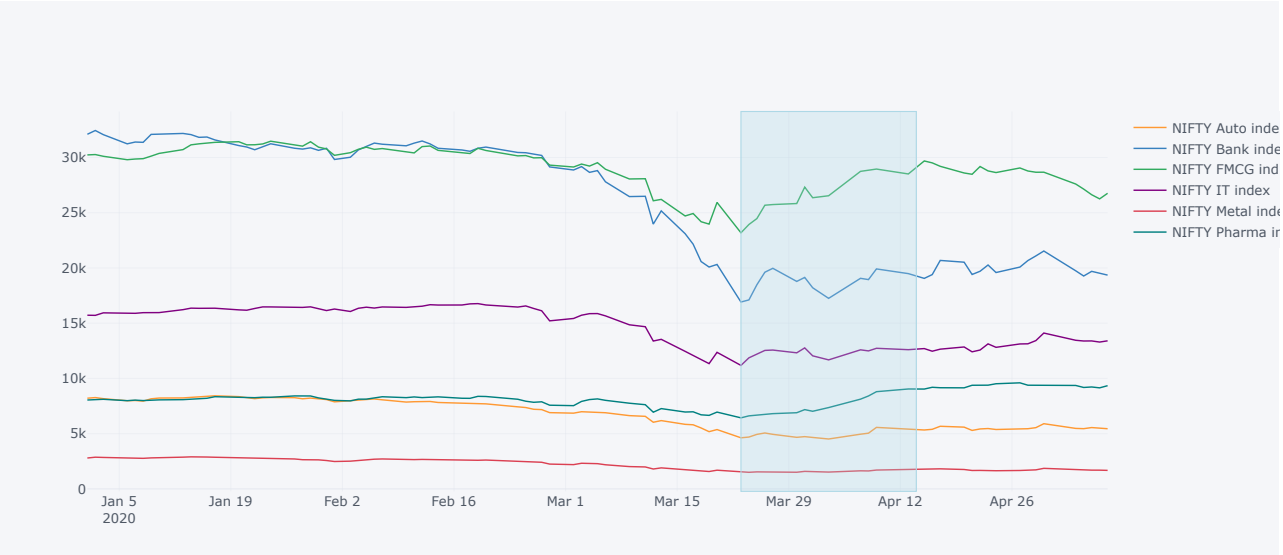
In [21]:
```
fig = df.iplot(asFigure = True, hline = [2,4], vline = ['2020-03-23'])
fig.show()
```



The red vertical line shows the day of the first phase of Lockdown in India.

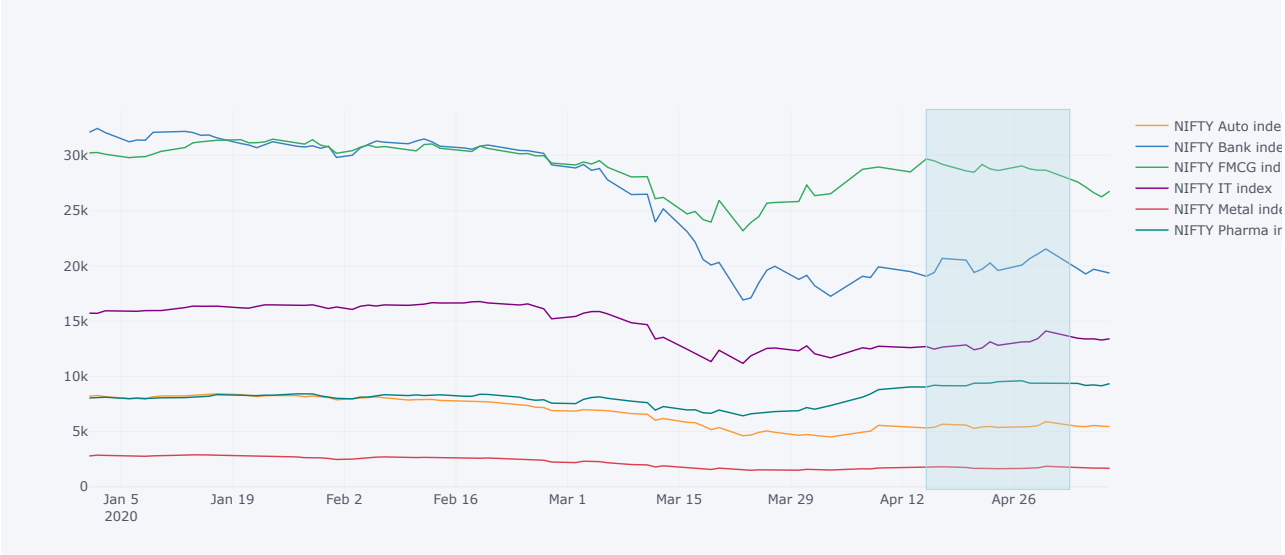
Lockdown Phase 1 (25 March – 14 April)

In [22]:
```
fig = df.iplot(asFigure=True,
               vspan={'x0':'2020-03-23','x1':'2020-04-14',
                      'color':'rgba(30,30,30,0.3)','color':'lightblue','fill':True,'opacity':0.3})
fig.show()
```



Lockdown Phase 2 (15 April – 3 May)

In [23]:
```
fig = df.iplot(asFigure=True,
               vspan={'x0':'2020-04-15','x1':'2020-05-03',
                      'color':'rgba(30,30,30,0.3)','color':'lightblue','fill':True,'opacity':0.3})
fig.show()
```

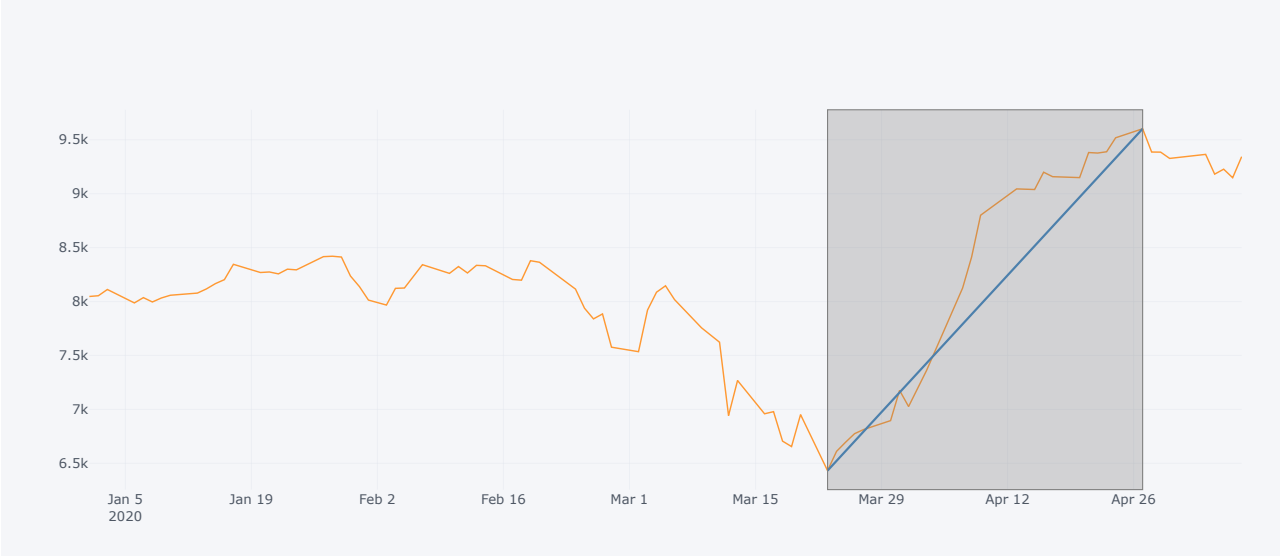


Better performing indices in 2020

NIFTY Pharma Index

In [24]:
```
df_a = df['NIFTY Pharma index']
max_val = df_a.max()
min_val = df_a.min()
max_date = df_a[df_a == max_val].index[0].strftime('%Y-%m-%d')
min_date = df_a[df_a == min_val].index[0].strftime('%Y-%m-%d')
shape1 = dict(kind = 'line', x0 = max_date, y0 = max_val, x1 = min_date, y1 = min_val, color = 'blue',width = 2)
shape2 = dict(kind = 'rect', x0 = max_date, x1 = min_date, fill = True, color = 'gray', opacity = 0.3)

df_a.iplot(shapes = [shape1, shape2])
```



The pharmaceutical sector experienced an initial dip during the first lockdown period due to COVID-19. However, it has since shown signs of recovery, driven by increased demand for healthcare products and strategic efforts by pharmaceutical companies to address challenges posed by the pandemic. Monitoring and adaptation remain essential for sustained progress.