In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
plt.style.use('ggplot')
%matplotlib inline
```

In [3]:
```
df = pd.read_excel('Dataset_JC.xlsx')
```

In [19]:
```
df.shape
```

Out[19]:
```
(119, 29)
```

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 119 entries, 0 to 118
Data columns (total 29 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   Date                   119 non-null    datetime64[ns]
 1   GSPC_Open              119 non-null    float64
 2   GSPC_High              119 non-null    float64
 3   GSPC_Low               119 non-null    float64
 4   GSPC_HighLow_price     119 non-null    float64
 5   GSPC_HighLow_percent   119 non-null    float64
 6   GSPC_Close             119 non-null    float64
 7   GSPC_AdjClose          119 non-null    float64
 8   GSPC_Volume            119 non-null    int64
 9   GSPC_Excess            118 non-null    float64
 10  DJI_Open               119 non-null    float64
 11  DJI_High               119 non-null    float64
 12  DJI_Low                119 non-null    float64
 13  DJI_HighLow_price      119 non-null    float64
 14  DJI_HighLow_percent    119 non-null    float64
 15  DJI_Close              119 non-null    float64
 16  DJI_AdjClose           119 non-null    float64
 17  DJI_Volume             119 non-null    int64
 18  DJI_Excess             118 non-null    float64
 19  FTS_Open               119 non-null    float64
 20  FTSE_High              119 non-null    float64
 21  FTSE_Low               119 non-null    float64
 22  FTSE_HighLow_price     119 non-null    float64
 23  FTSE_HighLow_price.1   119 non-null    float64
 24  FTSE_Close             119 non-null    float64
 25  FTSE_AdjClose          119 non-null    float64
 26  FTSE_Volume            119 non-null    int64
 27  CPI                    116 non-null    float64
 28  Unemployment           118 non-null    float64
dtypes: datetime64[ns](1), float64(25), int64(3)
memory usage: 27.1 KB
```

In [24]:
```
Numerical = df.iloc[:, 1:28]
```

In [25]:
```
Numerical
```

Out[25]:

	GSPC_Open	GSPC_High	GSPC_Low	GSPC_HighLow_price	GSPC_HighLow_percent	GSPC_C
0	1416.339966	1448.000000	1398.109985	49.890015	0.035225	1426.18
1	1426.189941	1509.939941	1426.189941	83.750000	0.058723	1498.10
2	1498.109985	1530.939941	1485.010010	45.929931	0.030659	1514.68
3	1514.680054	1570.280029	1501.479980	68.800049	0.045422	1569.18
4	1569.180054	1597.569946	1536.030029	61.539917	0.039218	1597.56
...	...	...	...	...	...	
114	4149.779785	4177.509766	3636.870117	540.639649	0.130282	3785.37
115	3781.000000	4140.149902	3721.560059	418.589843	0.110709	4130.29
116	4112.379883	4325.279785	3954.530029	370.749756	0.090155	3955.00
117	3936.729980	4119.279785	3584.129883	535.149902	0.135938	3585.62
118	3609.780029	3905.419922	3491.580078	413.839844	0.114644	3871.97

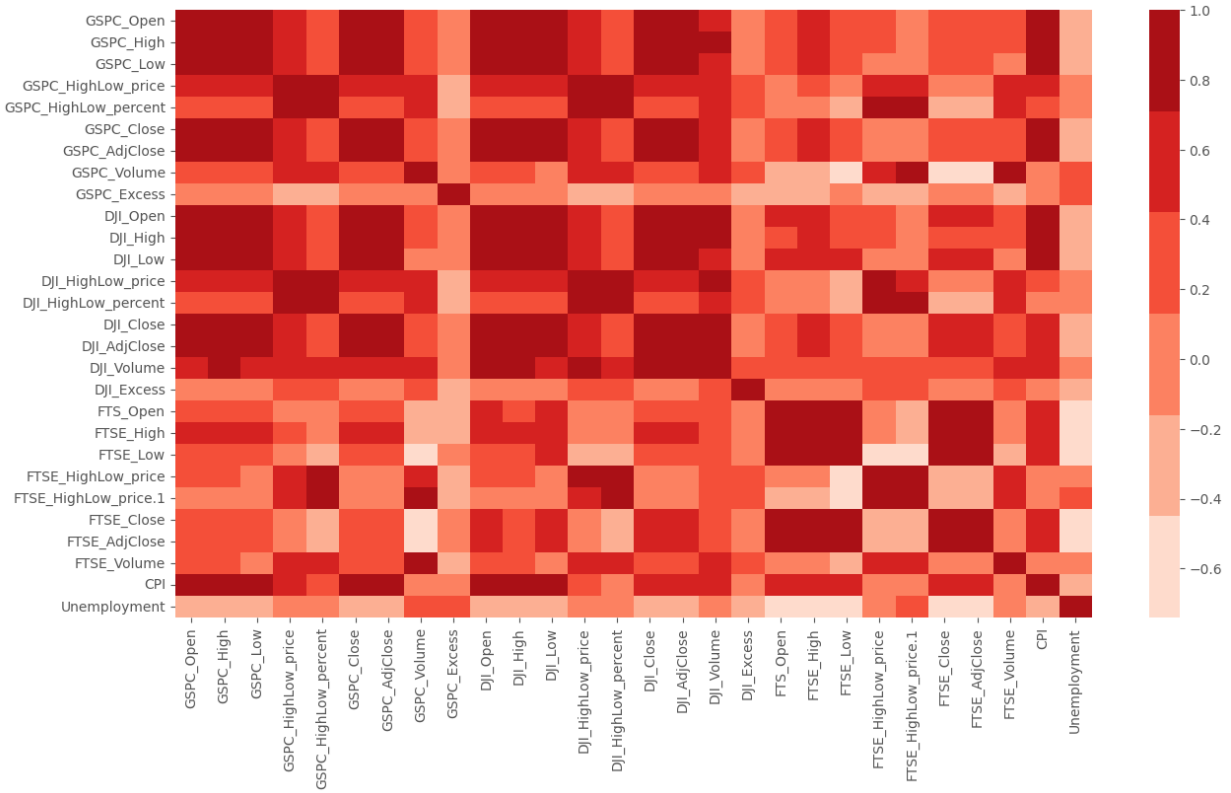
119 rows × 27 columns

In [26]:
```
import seaborn as sns
```

In [27]:
```
cormat = df.corr()
```

In [148]:
```
plt.figure(figsize = (15, 8))
sns.heatmap(cormat, cmap = sns.color_palette('Reds'))
```

Out[148]: <AxesSubplot: >



According to the correlation matrix, there are some groups of variables highly correlated, like ['GSPC_Open', 'GSPC_High', 'GSPC_Low', 'GSPC_Close', 'GSPC_Adf_Close', 'DJI_Open', 'DJI_High', 'DJI_Low', 'DJI_Close', 'DJI_AdjClose', 'CPI'], ['DJI_Volume', 'GSPC_High'], ['GSPC_HighLow_price', 'GSPC_HighLow_percent', 'DJI_HighLow_Price', 'DJI_HighLow_percent'] and so on, we focus ananlysis of these groups.

In [33]:
```
df.describe()
```

Out[33]:

	GSPC_Open	GSPC_High	GSPC_Low	GSPC_HighLow_price	GSPC_HighLow_percent	GSPC
count	119.000000	119.000000	119.000000	119.000000	119.000000	119.
mean	2697.850492	2789.711179	2605.892775	183.818403	0.064238	2715.
std	873.957824	913.609386	825.390400	150.809401	0.041503	869.
min	1416.339966	1448.000000	1398.109985	45.929931	0.019920	1426.
25%	2047.410034	2093.935059	1976.730041	78.280029	0.037004	2059.
50%	2498.080078	2657.739990	2443.959961	130.000000	0.053295	2575.
75%	3124.885010	3263.959961	3060.525024	225.954834	0.076384	3183.
max	4778.140137	4818.620117	4560.000000	944.859864	0.317677	4766.

8 rows × 28 columns

In [60]:
```
Numerical.columns[0]
```

Out[60]:
```
'GSPC_Open'
```

In [80]:
```
len([1, 2])
```

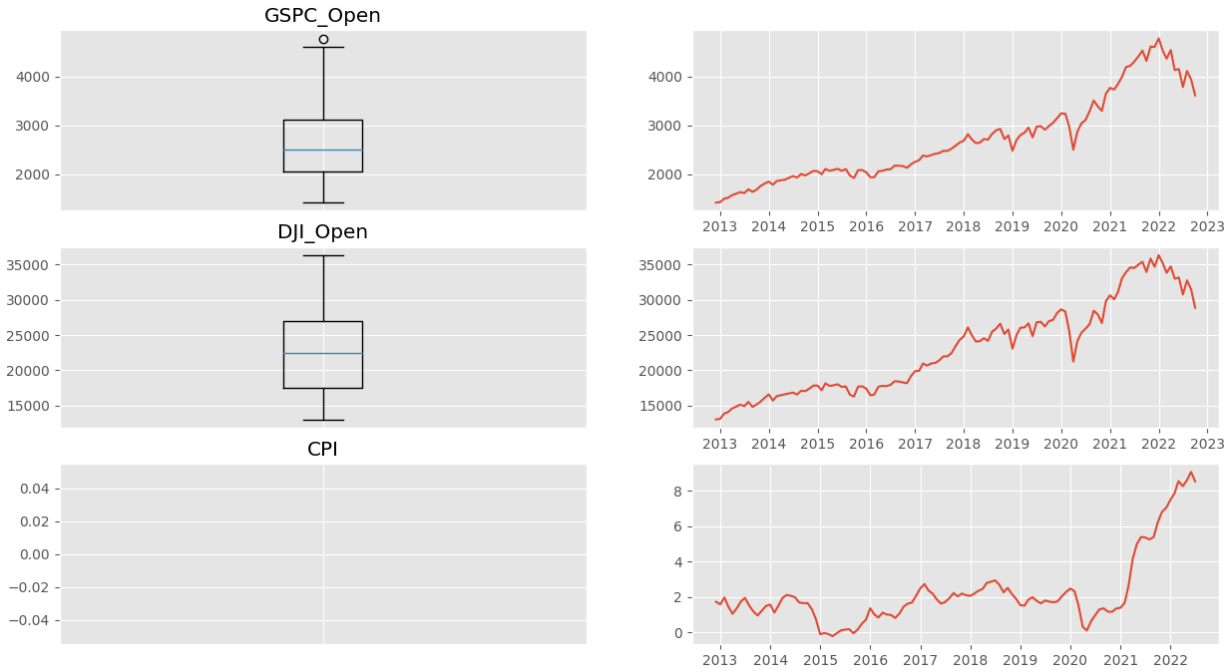
Out[80]:
```
2
```

In [118]:
```
def series_plot(group):
    length = len(group)
    fig, axs = plt.subplots(length, 2, figsize = (5 * length, 8), dpi = 100
    for i in range(0, length):
        axs[i, 0].boxplot(Numerical[group[i]].values)
        axs[i, 0].set_title(group[i])
        axs[i, 0].tick_params(axis = 'x', which = 'both', bottom = False, t
        axs[i, 1].plot(df['Date'], Numerical[group[i]].values)
```

Intuitively, the Highest price, Lowest price, open price and close price will change in the same way, as shown above. So we only keep one variable in the visualizations below to show the relationship between the variables.

In [121]:
```
group1 = ['GSPC_Open', 'DJI_Open', 'CPI']
```

In [122]:
```
series_plot(group1)
```

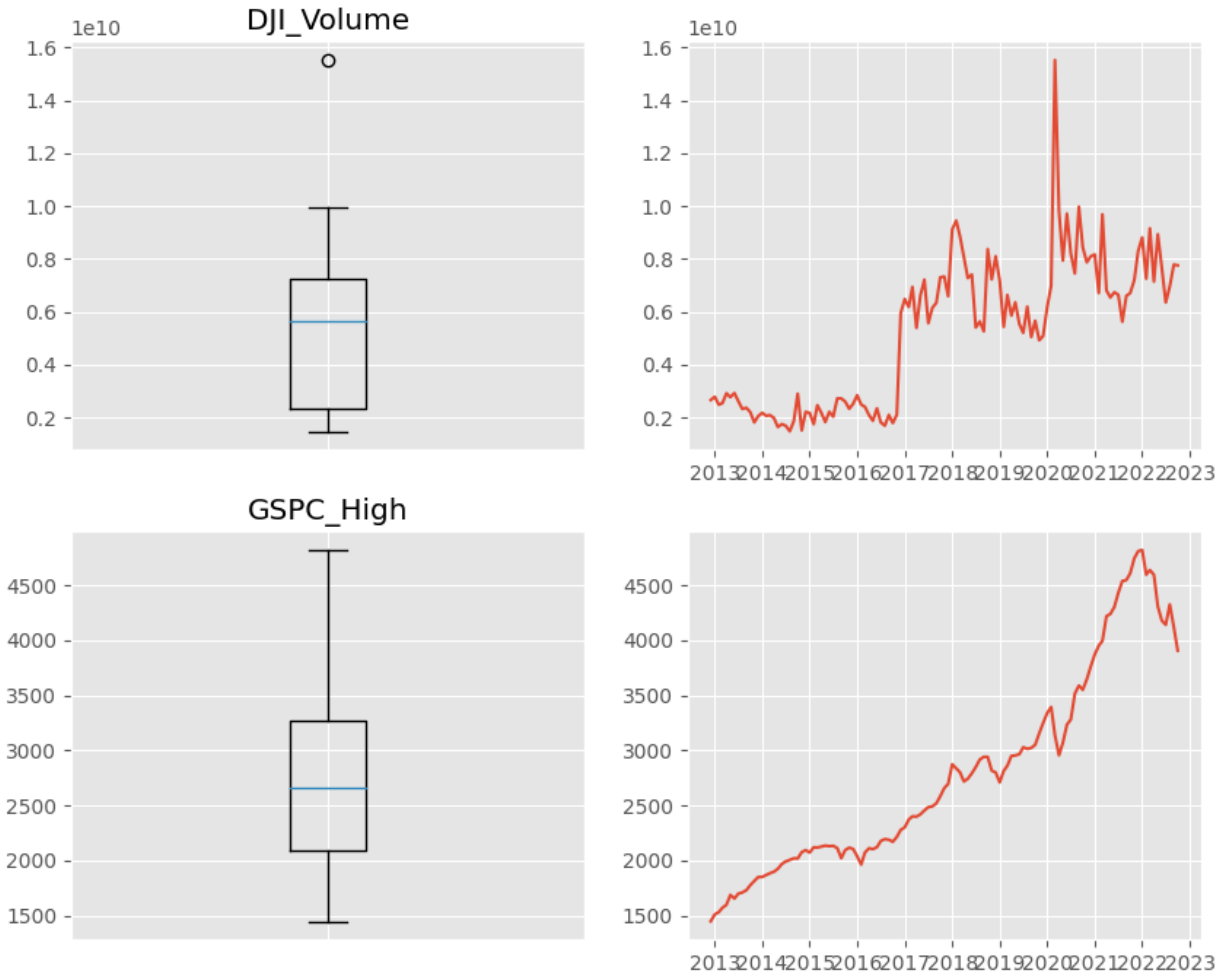


After 2020, the CPI rises quickly as well as the prices of the two indices, which leads to strong correlation. The CPI growth is the result of the fiscal stimulation and so is the indices, but the CPI has no strong causal relationship with stock market. According to the non-arbitrage property of the securities, the prices of component stocks should be exactly the same. And this is why the indices have similar trends.

Interestingly, the DJI_Volume is highly correlated with GSPC_High, and I want to use visualization to find the answer.

In [124]:
```
group2 = ['DJI_Volume', 'GSPC_High']
```

In [125]:
```
series_plot(group2)
```



In [135]:
```
df.loc[Numerical['DJI_Volume'].index[Numerical['DJI_Volume'] == Numerical['
```

Out[135]:
```
87   2020-03-01
Name: Date, dtype: datetime64[ns]
```

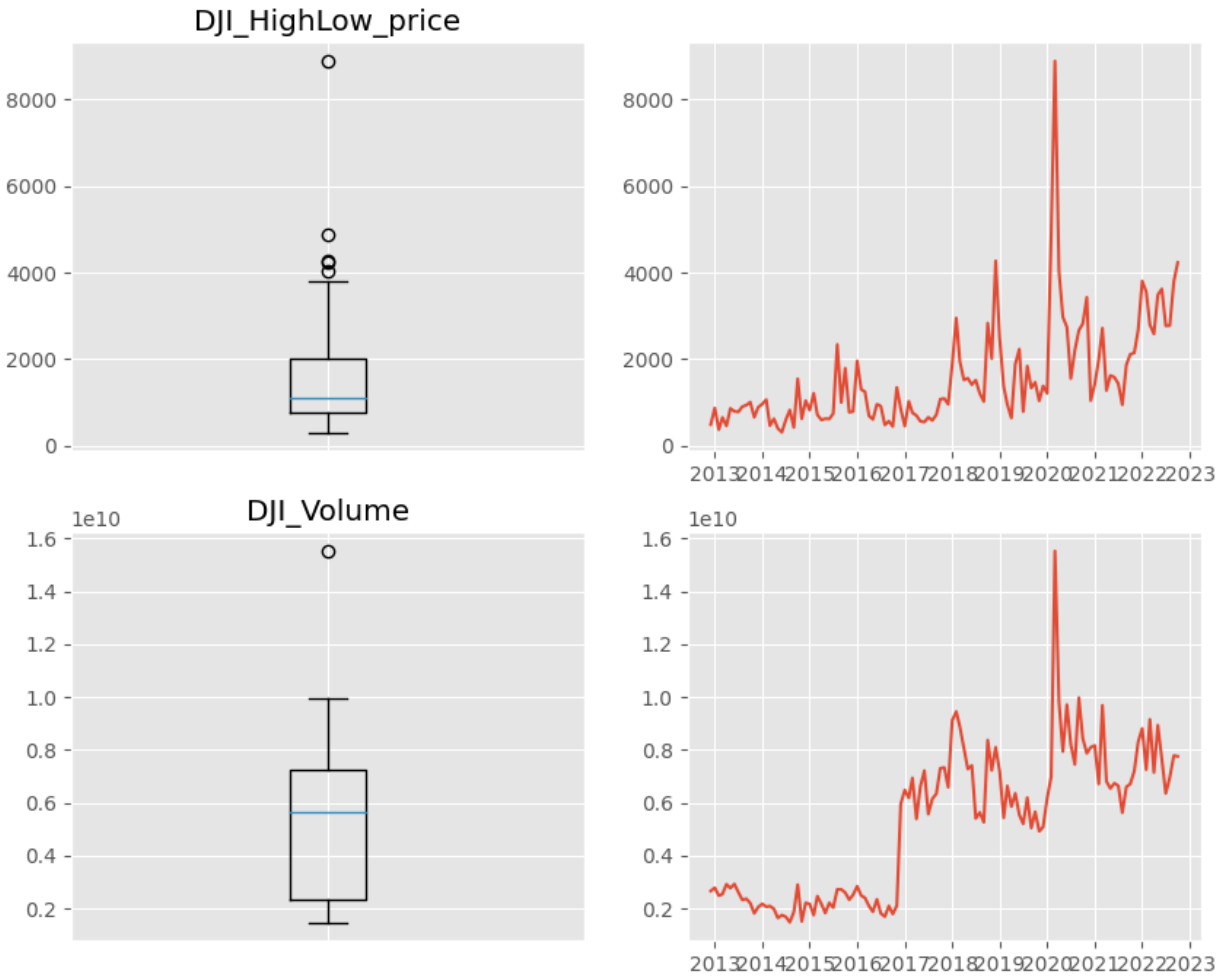
In [136]:
```
df.loc[Numerical['GSPC_High'].index[Numerical['GSPC_High'] == Numerical['GS
```

Out[136]:
```
109   2022-01-01
Name: Date, dtype: datetime64[ns]
```

After the outbreak of COVID 19 and the fiscal stimulations, market became hotter, and the speculaters became rasher, leading to higher daily volume, and it halted after Federal Reserve declared to rise the interest rate.

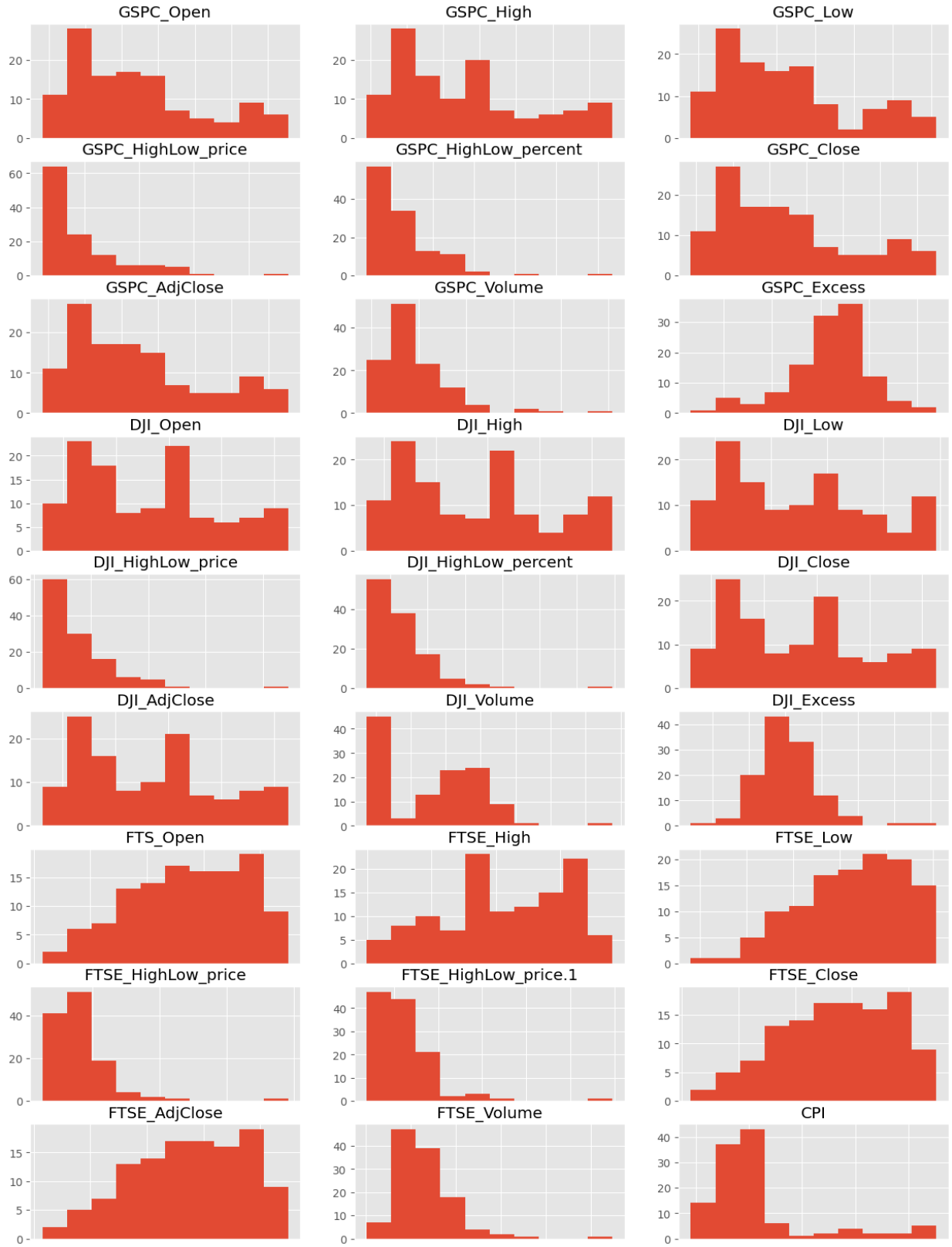
In [137]:
```
group3 = ['DJI_HighLow_price', 'DJI_Volume']
```

In [139]:
```
series_plot(group3)
```



Interestingly, the DJI_Volume and DJI_HighLow_price are highly correlated, this needs further investigation.

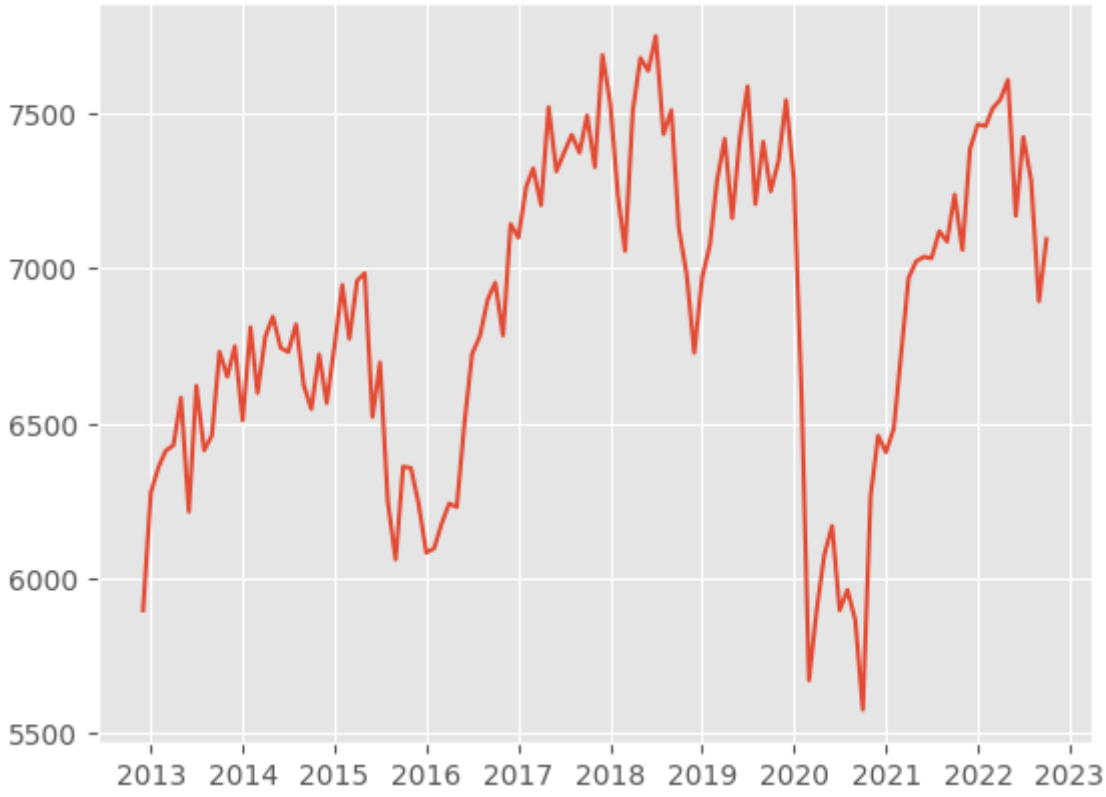
In [162]:
```
fig, axs = plt.subplots(9, 3, figsize = (15, 20), dpi = 100)
for i in range(0, 27):
    # fig = plt.figure(figsize = (, 3))
    axs[i//3, i%3].hist(Numerical[Numerical.columns[i]].values)
    axs[i//3, i%3].set_title(Numerical.columns[i])
    axs[i//3, i%3].tick_params(axis = 'x', which = 'both', bottom = False,
```



It seems that most of DJI and GSPC are right-skewed, and the distribution of FTSE is intriguing. Here we take FTSE_Close, which is left-skewed, into account.

In [149]:
```
plt.plot(df['Date'], df['FTSE_Close'])
```

Out[149]:
```
[<matplotlib.lines.Line2D at 0x1310bc070>]
```



The FTSE index is almost always high, except for the prevalance of COVID 19, namely, 2020-2021. And it returns to a normal level at the end of 2021

In [172]:
```
np.quantile(df['CPI'].dropna().values, [.1, .3, .5, .7, .9])
```

Out[172]:
```
array([0.41544713, 1.36666518, 1.73450041, 2.18814546, 5.37791205])
```

Also, CPI is concentrated at a high level, at around 1.73, indicating a mild inflation