In [1]:

```
import yfinance as yf

import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
```

In [2]:

```
sp_df = yf.download('^GSPC', start='2000-01-01', end='2014-12-31')
df = sp_df.loc[:, ['Adj Close']]
df = df.rename(columns={'Adj Close': 'Price'})
df.head(10)
```

```
[*********************100%***********************]  1 of 1 completed
```

Out[2]:

	Price
Date	
2000-01-03	1455.219971
2000-01-04	1399.420044
2000-01-05	1402.109985
2000-01-06	1403.449951
2000-01-07	1441.469971
2000-01-10	1457.599976
2000-01-11	1438.560059
2000-01-12	1432.250000
2000-01-13	1449.680054
2000-01-14	1465.150024

In [3]:

```
def get_vol(data, frequency):

    """
    function:calculate the annural volatility in an given frequency
    data:pandas dataframe or numpy array
    frequency:int, represent the sampling frequency in days
    """

    data = np.array(data)
    fprices = data[::frequency]
    log_return = np.log((fprices/np.roll(fprices,1))[1:])
    length_ = log_return.shape[0]
    delta_t = frequency/252

    mu = np.sum(log_return/length_)
    vol = np.sum(np.square(log_return-mu)/(length_-1))
    ann_sigma = np.sqrt(vol/delta_t)

    return ann_sigma
```

In [4]:

```
def calculate_exposure(df_, lookback=252):

    """
    function:calculate the exposure of the given strategy
    df_:dataframe with a column named Price
    lookback:int, days to lookback
    """

    df_['temp'] = 0
    daily_vol = get_vol(df_['Price'], frequency=1)
    weekly_vol = get_vol(df_['Price'], frequency=5)

    df_['temp'] = np.where(daily_vol > weekly_vol, 1,
                           np.where(daily_vol < weekly_vol, -1, 0))
    df_.iloc[:lookback, df_.columns.get_loc('temp')] = 0


    df_['First Day'] = df_.index - pd.to_timedelta(df_.index.dayofweek, unit='d')
    df_['First Day Price'] = df_.groupby('First Day')['Price'].transform('first')

    exposures = ((1 / df_['Price']) - (1 / df_['First Day Price'])) * df_['temp']

    return exposures.to_frame(name='Exposure')
```

In [5]:

```
def strategy_performance(cash, df, lookback=252):
    """
    function:calculate the performance of the strategy
    """
    df_ = calculate_exposure(df, lookback=lookback)

    df_['Daily P/L'] = cash * df_['Exposure'].shift(1) * (df['Price'] - df['Price'
    df_['Cumulative P/L'] = df_['Daily P/L'].cumsum()

    df_['Return'] = df_['Cumulative P/L'].pct_change()
    df_ = df_.replace(np.inf,0)
    df_['Cumulative Return'] = (1 + df_['Return']).cumprod() - 1

    return df_.drop(['Daily P/L','Return'], axis=1)
```

In [6]:

```
performance = strategy_performance(10000, df, lookback=252)
performance
```

Out[6]:

	Exposure	Cumulative P/L	Cumulative Return
Date			
2000-01-03	0.000000e+00	NaN	NaN
2000-01-04	0.000000e+00	0.000000	NaN
2000-01-05	0.000000e+00	0.000000	NaN
2000-01-06	0.000000e+00	0.000000	NaN
2000-01-07	0.000000e+00	0.000000	NaN
...	...	...	...
2014-12-23	-8.387220e-07	225.994552	41.748219
2014-12-24	-7.718130e-07	225.996985	41.748679
2014-12-26	-2.356274e-06	225.943806	41.738620
2014-12-29	0.000000e+00	225.901392	41.730598
2014-12-30	2.349895e-06	225.901392	41.730598

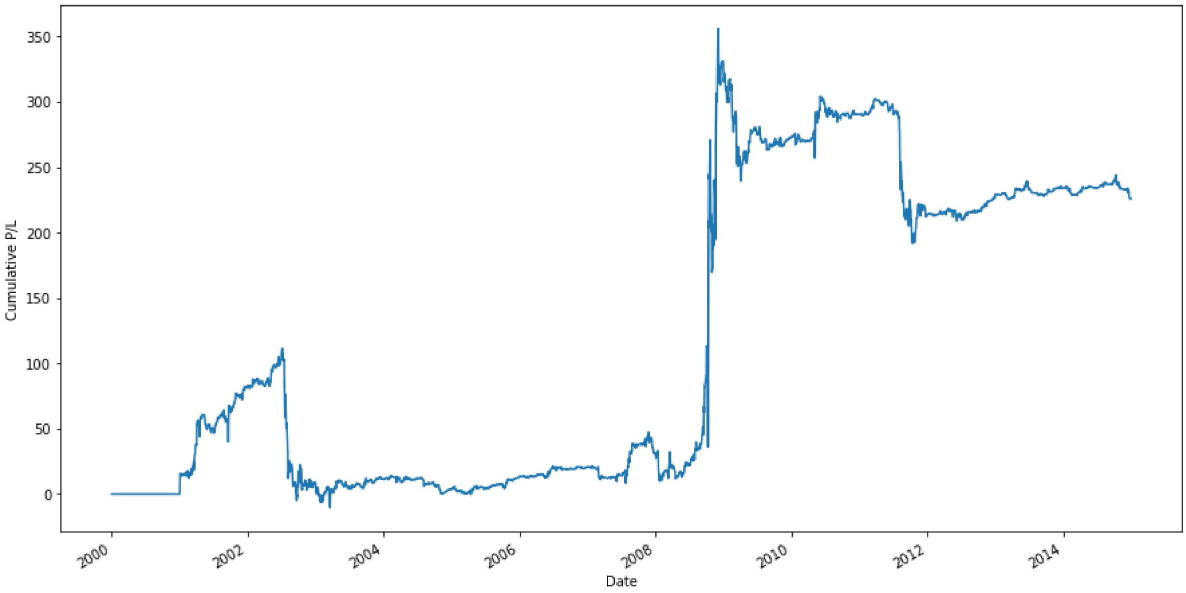
3772 rows × 3 columns

In [7]:

```
fig = plt.figure(figsize=(15, 8))
```

```
performance['Cumulative P/L'].plot()
plt.ylabel("Cumulative P/L")

plt.show()
```



```
In [ ]:
```