In [1]:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import datetime
```

In [3]:

```
dataset=pd.read_csv('Google_Stock_Price_Train.csv',index_col="Date",parse_dates=True)
```

In [4]:

```
dataset.head()
```

Out[4]:

	Open	High	Low	Close	Volume
Date					
2012-01-03	325.25	332.83	324.97	663.59	7,380,500
2012-01-04	331.27	333.87	329.08	666.45	5,749,400
2012-01-05	329.83	330.75	326.89	657.21	6,590,300
2012-01-06	328.34	328.77	323.68	648.24	5,405,900
2012-01-09	322.04	322.29	309.46	620.76	11,688,800

In [5]:

```
dataset.isna().any()
```

Out[5]:

```
Open      False
High      False
Low       False
Close     False
Volume    False
dtype: bool
```

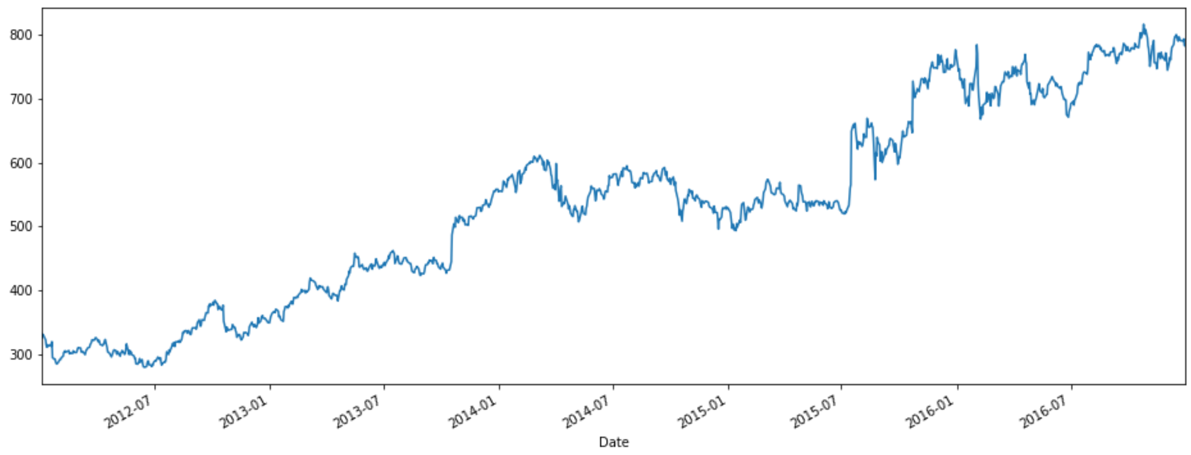
In [6]:

```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 1258 entries, 2012-01-03 to 2016-12-30
Data columns (total 5 columns):
Open      1258 non-null float64
High      1258 non-null float64
Low       1258 non-null float64
Close     1258 non-null object
Volume    1258 non-null object
dtypes: float64(3), object(2)
memory usage: 59.0+ KB
```

```
In [8]: dataset['Open'].plot(figsize=(16,6))
```

```
Out[8]: <matplotlib.axes._subplots.AxesSubplot at 0xd191696978>
```



```
In [10]: #convert column a of a DataFrame
         dataset["Close"] = dataset["Close"].str.replace(',', '').astype(float)
```

```
In [11]: dataset["Volume"] = dataset["Volume"].str.replace(',', '').astype(float)
```

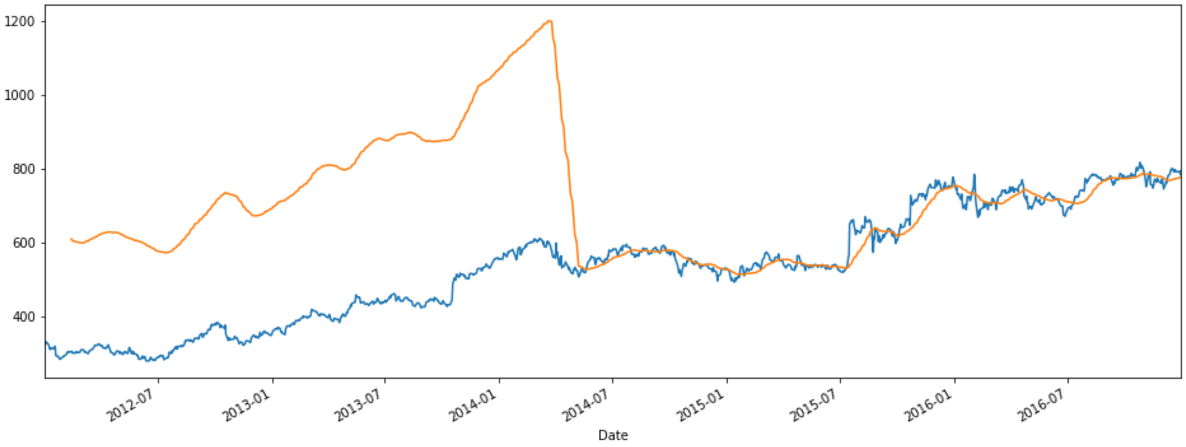
In [14]:
```
dataset.rolling(7).mean().head(25)
```

Out[14]:

	Open	High	Low	Close	Volume
Date					
2012-01-03	NaN	NaN	NaN	NaN	NaN
2012-01-04	NaN	NaN	NaN	NaN	NaN
2012-01-05	NaN	NaN	NaN	NaN	NaN
2012-01-06	NaN	NaN	NaN	NaN	NaN
2012-01-09	NaN	NaN	NaN	NaN	NaN
2012-01-10	NaN	NaN	NaN	NaN	NaN
2012-01-11	323.002857	325.392857	318.682857	643.132857	7.208100e+06
2012-01-12	321.457143	322.882857	316.841429	638.037143	6.691514e+06
2012-01-13	318.698571	319.801429	314.025714	631.870000	6.531857e+06
2012-01-17	316.552857	317.524286	311.851429	627.534286	6.137929e+06
2012-01-18	314.238571	315.674286	309.882857	625.097143	6.157657e+06
2012-01-19	313.847143	315.247143	310.610000	627.534286	6.296086e+06
2012-01-20	311.055714	312.201429	308.104286	622.242857	8.068629e+06
2012-01-23	308.387143	309.302857	305.402857	616.481429	8.359129e+06
2012-01-24	305.192857	306.085714	301.951429	609.541429	8.697700e+06
2012-01-25	301.724286	302.652857	298.060000	601.634286	9.466400e+06
2012-01-26	297.454286	298.561429	293.710000	593.017143	9.844071e+06
2012-01-27	293.480000	294.741429	289.952857	585.475714	1.008950e+07
2012-01-30	289.001429	290.401429	285.821429	576.660000	8.949586e+06
2012-01-31	288.465714	289.902857	285.355714	575.821429	6.530857e+06
2012-02-01	288.390000	289.678571	285.070000	575.152857	6.217629e+06
2012-02-02	288.285714	289.588571	285.360000	575.748571	6.033771e+06
2012-02-03	289.221429	290.895714	286.902857	579.572857	5.512057e+06
2012-02-06	290.860000	293.481429	289.000000	585.412857	5.642086e+06
2012-02-07	293.448571	295.550000	291.450000	589.230000	5.204614e+06

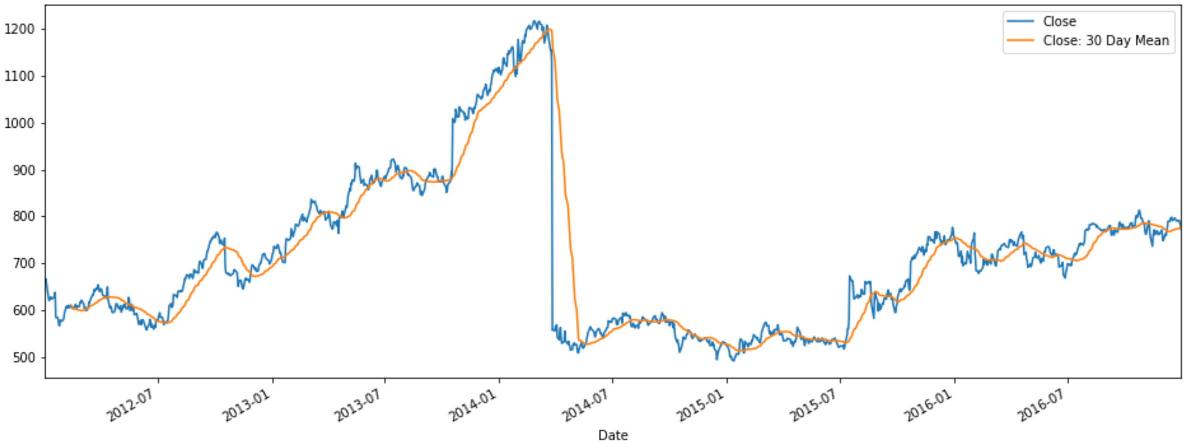
In [15]:
```
dataset['Open'].plot(figsize=(16,6))
dataset.rolling(window=30).mean()['Close'].plot()
```

Out[15]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xd193124b70>
```



In [16]:
```
dataset['Close: 30 Day Mean'] = dataset['Close'].rolling(window=30).mean()
dataset[['Close','Close: 30 Day Mean']].plot(figsize=(16,6))
```

Out[16]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xd193176588>
```

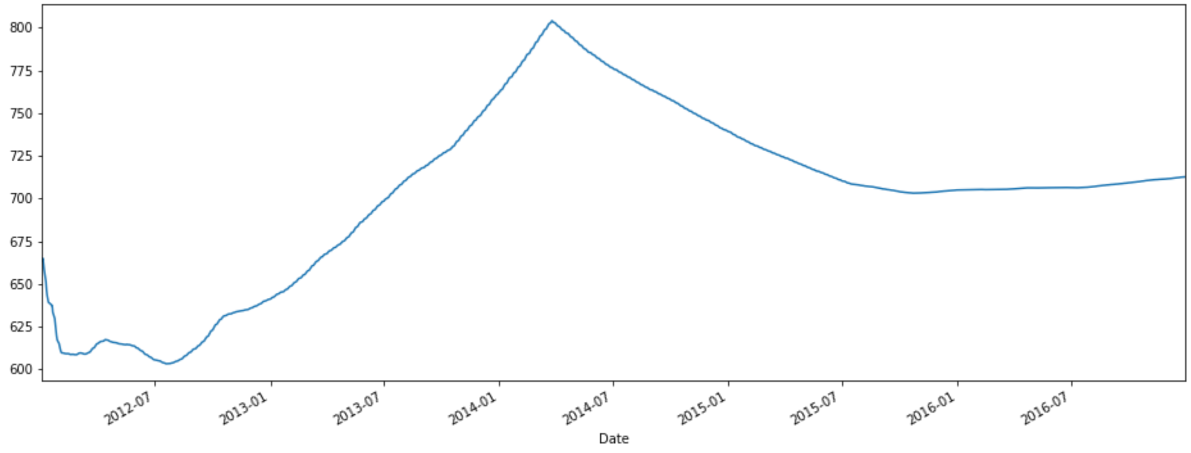


In [17]:

```
#Optional specify minimum number of periods
dataset['Close'].expanding(min_periods=1).mean().plot(figsize=(16,6))
```

Out[17]:

```
<matplotlib.axes._subplots.AxesSubplot at 0xd1917cbd30>
```



In [18]:

```
training_set = dataset['Open']
training_set = pd.DataFrame(training_set)
```

In [19]:

```
#Feature Scaling
from sklearn.preprocessing import MinMaxScaler
sc = MinMaxScaler(feature_range = (0,1))
training_set_scaled = sc.fit_transform(training_set)
```

In [20]:

```
#Creating a data structure with 60 timesteps and 1 output
X_train = []
Y_train = []
for i in range(60,1258):
    X_train.append(training_set_scaled[i-60:i, 0])
    Y_train.append(training_set_scaled[i, 0])
X_train, Y_train = np.array(X_train), np.array(Y_train)

#Reshaping
X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))
```

In [21]:

```
# RNN Model

from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from keras.layers import Dropout
```

```
Using TensorFlow backend.
```

In [22]:

```
regressor = Sequential()
```

In [24]:

```
#Adding the first LSTM layer and some Dropout regularisation
regressor.add(LSTM(units=50, return_sequences=True, input_shape = (X_train.shape[1], 1)))
regressor.add(Dropout(0.2))

#Adding a second LSTM layer and some Dropout regularisation
regressor.add(LSTM(units=50,return_sequences=True))
regressor.add(Dropout(0.2))

#Adding a third LSTM layer and some Dropout regularisation
regressor.add(LSTM(units=50, return_sequences=True))
regressor.add(Dropout(0.2))

#Adding a fourth LSTM layer and some Dropout regularisation
regressor.add(LSTM(units=50))

#Adding the output layer
regressor.add(Dense(units=1))
```

In [25]:

```
#Compiling the RNN
regressor.compile(optimizer='adam', loss='mean_squared_error')

#Fitting the RNN to the training set
regressor.fit(X_train, Y_train, epochs=100, batch_size=32)
```

```
Epoch 1/100
1198/1198 [==============================] - 9s 8ms/step - loss: 0.0407
Epoch 2/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0032
Epoch 3/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0026
Epoch 4/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0026
Epoch 5/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0025
Epoch 6/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0028
Epoch 7/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0022
Epoch 8/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0024
Epoch 9/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0024
Epoch 10/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0023
Epoch 11/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0021
Epoch 12/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0026
Epoch 13/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0020
Epoch 14/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0020
Epoch 15/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0023
Epoch 16/100
1198/1198 [==============================] - 5s 5ms/step - loss: 0.0020
Epoch 17/100
1198/1198 [==============================] - 6s 5ms/step - loss: 0.0020
Epoch 18/100
1198/1198 [==============================] - 5s 5ms/step - loss: 0.0018
Epoch 19/100
1198/1198 [==============================] - 5s 5ms/step - loss: 0.0018
Epoch 20/100
1198/1198 [==============================] - 6s 5ms/step - loss: 0.0018
Epoch 21/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0019
Epoch 22/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0017
Epoch 23/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0021
Epoch 24/100
1198/1198 [==============================] - 5s 5ms/step - loss: 0.0021
Epoch 25/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0016
Epoch 26/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0017
Epoch 27/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0018
Epoch 28/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0018
Epoch 29/100
```

```
1198/1198 [==============================] - 5s 5ms/step - loss: 0.0017
Epoch 30/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0015
Epoch 31/100
1198/1198 [==============================] - 6s 5ms/step - loss: 0.0015
Epoch 32/100
1198/1198 [==============================] - 6s 5ms/step - loss: 0.0015
Epoch 33/100
1198/1198 [==============================] - 6s 5ms/step - loss: 0.0017
Epoch 34/100
1198/1198 [==============================] - 6s 5ms/step - loss: 0.0016
Epoch 35/100
1198/1198 [==============================] - 6s 5ms/step - loss: 0.0014
Epoch 36/100
1198/1198 [==============================] - 6s 5ms/step - loss: 0.0015
Epoch 37/100
1198/1198 [==============================] - 6s 5ms/step - loss: 0.0016
Epoch 38/100
1198/1198 [==============================] - 5s 5ms/step - loss: 0.0014
Epoch 39/100
1198/1198 [==============================] - 6s 5ms/step - loss: 0.0014
Epoch 40/100
1198/1198 [==============================] - 6s 5ms/step - loss: 0.0013
Epoch 41/100
1198/1198 [==============================] - 6s 5ms/step - loss: 0.0013
Epoch 42/100
1198/1198 [==============================] - 5s 5ms/step - loss: 0.0014
Epoch 43/100
1198/1198 [==============================] - 5s 5ms/step - loss: 0.0013
Epoch 44/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0012
Epoch 45/100
1198/1198 [==============================] - 5s 5ms/step - loss: 0.0012
Epoch 46/100
1198/1198 [==============================] - 5s 5ms/step - loss: 0.0012
Epoch 47/100
1198/1198 [==============================] - 5s 5ms/step - loss: 0.0014
Epoch 48/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0013
Epoch 49/100
1198/1198 [==============================] - 5s 5ms/step - loss: 0.0012
Epoch 50/100
1198/1198 [==============================] - 6s 5ms/step - loss: 0.0012
Epoch 51/100
1198/1198 [==============================] - 6s 5ms/step - loss: 0.0012
Epoch 52/100
1198/1198 [==============================] - 5s 5ms/step - loss: 0.0013
Epoch 53/100
1198/1198 [==============================] - 5s 5ms/step - loss: 0.0013
Epoch 54/100
1198/1198 [==============================] - 5s 5ms/step - loss: 0.0013
Epoch 55/100
1198/1198 [==============================] - 6s 5ms/step - loss: 0.0012
Epoch 56/100
1198/1198 [==============================] - 6s 5ms/step - loss: 0.0011
Epoch 57/100
1198/1198 [==============================] - 5s 5ms/step - loss: 0.0012
```

```
Epoch 58/100
1198/1198 [==============================] - 6s 5ms/step - loss: 0.0012
Epoch 59/100
1198/1198 [==============================] - 5s 5ms/step - loss: 0.0011
Epoch 60/100
1198/1198 [==============================] - 5s 5ms/step - loss: 9.8752e-04
Epoch 61/100
1198/1198 [==============================] - 6s 5ms/step - loss: 0.0010
Epoch 62/100
1198/1198 [==============================] - 6s 5ms/step - loss: 0.0012
Epoch 63/100
1198/1198 [==============================] - 5s 5ms/step - loss: 0.0012
Epoch 64/100
1198/1198 [==============================] - 5s 5ms/step - loss: 0.0012
Epoch 65/100
1198/1198 [==============================] - 5s 4ms/step - loss: 9.8793e-04
Epoch 66/100
1198/1198 [==============================] - 5s 5ms/step - loss: 0.0010
Epoch 67/100
1198/1198 [==============================] - 5s 5ms/step - loss: 9.0990e-04
Epoch 68/100
1198/1198 [==============================] - 5s 4ms/step - loss: 9.6066e-04
Epoch 69/100
1198/1198 [==============================] - 5s 4ms/step - loss: 8.9573e-04
Epoch 70/100
1198/1198 [==============================] - 6s 5ms/step - loss: 8.2925e-04
Epoch 71/100
1198/1198 [==============================] - 5s 4ms/step - loss: 8.8818e-04
Epoch 72/100
1198/1198 [==============================] - 5s 5ms/step - loss: 8.3686e-04
Epoch 73/100
1198/1198 [==============================] - 5s 5ms/step - loss: 8.0439e-04
Epoch 74/100
1198/1198 [==============================] - 6s 5ms/step - loss: 8.3554e-04
Epoch 75/100
1198/1198 [==============================] - 6s 5ms/step - loss: 8.2431e-04
Epoch 76/100
1198/1198 [==============================] - 6s 5ms/step - loss: 8.3949e-04
Epoch 77/100
1198/1198 [==============================] - 6s 5ms/step - loss: 8.4586e-04
Epoch 78/100
1198/1198 [==============================] - 5s 5ms/step - loss: 7.6545e-04
Epoch 79/100
1198/1198 [==============================] - 5s 5ms/step - loss: 7.5509e-04
Epoch 80/100
1198/1198 [==============================] - 5s 4ms/step - loss: 0.0011
Epoch 81/100
1198/1198 [==============================] - 6s 5ms/step - loss: 7.9825e-04
Epoch 82/100
1198/1198 [==============================] - 5s 5ms/step - loss: 8.3659e-04
Epoch 83/100
1198/1198 [==============================] - 5s 4ms/step - loss: 8.5187e-04
Epoch 84/100
1198/1198 [==============================] - 5s 4ms/step - loss: 7.9683e-04
Epoch 85/100
1198/1198 [==============================] - 5s 4ms/step - loss: 7.1779e-04
Epoch 86/100
```

```
1198/1198 [==============================] - 5s 4ms/step - loss: 8.2978e-04
Epoch 87/100
1198/1198 [==============================] - 5s 5ms/step - loss: 0.0012
Epoch 88/100
1198/1198 [==============================] - 5s 4ms/step - loss: 6.8000e-04
Epoch 89/100
1198/1198 [==============================] - 5s 4ms/step - loss: 8.5700e-04
Epoch 90/100
1198/1198 [==============================] - 5s 4ms/step - loss: 7.1847e-04
Epoch 91/100
1198/1198 [==============================] - 5s 5ms/step - loss: 7.2458e-04
Epoch 92/100
1198/1198 [==============================] - 5s 4ms/step - loss: 8.8821e-04
Epoch 93/100
1198/1198 [==============================] - 5s 4ms/step - loss: 7.3291e-04
Epoch 94/100
1198/1198 [==============================] - 5s 4ms/step - loss: 7.0305e-04
Epoch 95/100
1198/1198 [==============================] - 5s 4ms/step - loss: 6.9923e-04
Epoch 96/100
1198/1198 [==============================] - 5s 4ms/step - loss: 7.1799e-04
Epoch 97/100
1198/1198 [==============================] - 5s 4ms/step - loss: 6.8517e-04
Epoch 98/100
1198/1198 [==============================] - 5s 4ms/step - loss: 7.7327e-04
Epoch 99/100
1198/1198 [==============================] - 5s 4ms/step - loss: 6.3169e-04
Epoch 100/100
1198/1198 [==============================] - 5s 4ms/step - loss: 6.2063e-04
```

Out[25]:

```
<keras.callbacks.History at 0xd1a4e9df98>
```

In [28]:

```
dataset_test = pd.read_csv('Google_Stock_Price_Test.csv',index_col="Date",parse_dates=True)
```

In [29]:

```
real_stock_price = dataset_test.iloc[:, 1:2].values
```

In [30]:

```
dataset_test.head()
```

Out[30]:

	Open	High	Low	Close	Volume
Date					
2017-01-03	778.81	789.63	775.80	786.14	1,657,300
2017-01-04	788.36	791.34	783.16	786.90	1,073,000
2017-01-05	786.08	794.48	785.02	794.02	1,335,200
2017-01-06	795.26	807.90	792.20	806.15	1,640,200
2017-01-09	806.40	809.97	802.83	806.65	1,272,400

In [31]:

```
dataset_test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 20 entries, 2017-01-03 to 2017-01-31
Data columns (total 5 columns):
Open      20 non-null float64
High      20 non-null float64
Low       20 non-null float64
Close     20 non-null float64
Volume    20 non-null object
dtypes: float64(4), object(1)
memory usage: 960.0+ bytes
```

In [32]:

```
dataset_test["Volume"] = dataset_test["Volume"].str.replace(',', '').astype(fl
oat)
```

In [33]:

```
test_set = dataset_test['Open']
test_set = pd.DataFrame(test_set)
```

In [34]:

```
test_set.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 20 entries, 2017-01-03 to 2017-01-31
Data columns (total 1 columns):
Open    20 non-null float64
dtypes: float64(1)
memory usage: 320.0 bytes
```

In [36]:

```
dataset_total = pd.concat((dataset['Open'], dataset_test['Open']
                          ), axis=0)
inputs = dataset_total[len(dataset_total) - len(dataset_test) - 60:].values
inputs = inputs.reshape(-1,1)
inputs = sc.transform(inputs)
X_test = []
for i in range(60, 80):
    X_test.append(inputs[i-60:i, 0])
X_test = np.array(X_test)
X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
predicted_stock_price = regressor.predict(X_test)
predicted_stock_price = sc.inverse_transform(predicted_stock_price)
```

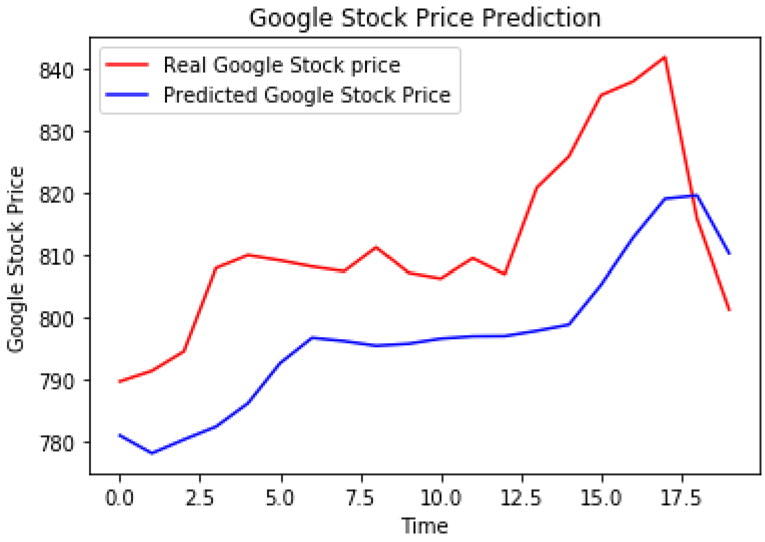
In [37]:

```
predicted_stock_price=pd.DataFrame(predicted_stock_price)
predicted_stock_price.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20 entries, 0 to 19
Data columns (total 1 columns):
0    20 non-null float32
dtypes: float32(1)
memory usage: 160.0 bytes
```

In [38]:

```
plt.plot(real_stock_price, color='red', label='Real Google Stock price')
plt.plot(predicted_stock_price, color = 'blue', label = 'Predicted Google Stoc
k Price')
plt.title('Google Stock Price Prediction')
plt.xlabel('Time')
plt.ylabel('Google Stock Price')
plt.legend()
plt.show()
```



In [ ]: