In [5]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
dataset_train = pd.read_csv('Google_Stock_Price_Train.csv')
print(dataset_train)
```

```
            Date    Open    High     Low   Close      Volume
0       1/3/2012  325.25  332.83  324.97  663.59   7,380,500
1       1/4/2012  331.27  333.87  329.08  666.45   5,749,400
2       1/5/2012  329.83  330.75  326.89  657.21   6,590,300
3       1/6/2012  328.34  328.77  323.68  648.24   5,405,900
4       1/9/2012  322.04  322.29  309.46  620.76  11,688,800
...          ...     ...     ...     ...     ...         ...
1253  12/23/2016  790.90  792.74  787.28  789.91     623,400
1254  12/27/2016  790.68  797.86  787.66  791.55     789,100
1255  12/28/2016  793.70  794.23  783.20  785.05   1,153,800
1256  12/29/2016  783.33  785.93  778.92  782.79     744,300
1257  12/30/2016  782.75  782.78  770.41  771.82   1,770,000

[1258 rows x 6 columns]
```

In [6]:
```
dataset_train = pd.read_csv('Google_Stock_Price_Train.csv')
training_set = dataset_train.iloc[:, 1:2].values
print(training_set)
```

```
[[325.25]
 [331.27]
 [329.83]
 ...
 [793.7 ]
 [783.33]
 [782.75]]
```

In [7]:
```
from sklearn.preprocessing import MinMaxScaler
sc = MinMaxScaler(feature_range = (0, 1))
training_set_scaled = sc.fit_transform(training_set)
print(training_set_scaled)
```

```
[[0.08581368]
 [0.09701243]
 [0.09433366]
 ...
 [0.95725128]
 [0.93796041]
 [0.93688146]]
```

In [11]:
```
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)

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

In [12]:
```
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from keras.layers import Dropout

# Initialising the RNN
regressor = Sequential()

# 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))
regressor.add(Dropout(0.2))

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

# 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
38/38 [==============================] - 18s 130ms/step - loss: 0.0314
Epoch 2/100
38/38 [==============================] - 4s 108ms/step - loss: 0.0065
Epoch 3/100
38/38 [==============================] - 4s 105ms/step - loss: 0.0052
Epoch 4/100
38/38 [==============================] - 4s 110ms/step - loss: 0.0054
Epoch 5/100
38/38 [==============================] - 4s 98ms/step - loss: 0.0058
Epoch 6/100
38/38 [==============================] - 4s 97ms/step - loss: 0.0047
Epoch 7/100
38/38 [==============================] - 5s 120ms/step - loss: 0.0060
Epoch 8/100
38/38 [==============================] - 4s 98ms/step - loss: 0.0046
Epoch 9/100
38/38 [==============================] - 4s 110ms/step - loss: 0.0042
Epoch 10/100
38/38 [==============================] - 4s 115ms/step - loss: 0.0043
Epoch 11/100
38/38 [==============================] - 4s 106ms/step - loss: 0.0045
Epoch 12/100
38/38 [==============================] - 4s 118ms/step - loss: 0.0048
Epoch 13/100
38/38 [==============================] - 4s 97ms/step - loss: 0.0038
Epoch 14/100
38/38 [==============================] - 4s 119ms/step - loss: 0.0042
Epoch 15/100
38/38 [==============================] - 4s 106ms/step - loss: 0.0040
Epoch 16/100
38/38 [==============================] - 5s 126ms/step - loss: 0.0040
Epoch 17/100
38/38 [==============================] - 4s 109ms/step - loss: 0.0038
Epoch 18/100
38/38 [==============================] - 5s 122ms/step - loss: 0.0036
Epoch 19/100
38/38 [==============================] - 4s 104ms/step - loss: 0.0039
Epoch 20/100
38/38 [==============================] - 4s 108ms/step - loss: 0.0035
Epoch 21/100
38/38 [==============================] - 4s 115ms/step - loss: 0.0036
Epoch 22/100
38/38 [==============================] - 5s 125ms/step - loss: 0.0040
Epoch 23/100
38/38 [==============================] - 4s 96ms/step - loss: 0.0033
Epoch 24/100
38/38 [==============================] - 4s 116ms/step - loss: 0.0033
Epoch 25/100
38/38 [==============================] - 5s 126ms/step - loss: 0.0031
Epoch 26/100
38/38 [==============================] - 5s 120ms/step - loss: 0.0035
Epoch 27/100
38/38 [==============================] - 4s 101ms/step - loss: 0.0031
Epoch 28/100
38/38 [==============================] - 4s 117ms/step - loss: 0.0032
Epoch 29/100
38/38 [==============================] - 5s 121ms/step - loss: 0.0030
Epoch 30/100
38/38 [==============================] - 5s 126ms/step - loss: 0.0025
Epoch 31/100
38/38 [==============================] - 4s 111ms/step - loss: 0.0026
Epoch 32/100
38/38 [==============================] - 4s 117ms/step - loss: 0.0029
Epoch 33/100
38/38 [==============================] - 4s 112ms/step - loss: 0.0029
Epoch 34/100
38/38 [==============================] - 5s 120ms/step - loss: 0.0030
Epoch 35/100
38/38 [==============================] - 4s 110ms/step - loss: 0.0027
Epoch 36/100
38/38 [==============================] - 5s 123ms/step - loss: 0.0026
Epoch 37/100
38/38 [==============================] - 4s 112ms/step - loss: 0.0025
Epoch 38/100
```

```
38/38 [==============================] - 5s 117ms/step - loss: 0.0028
Epoch 39/100
38/38 [==============================] - 4s 113ms/step - loss: 0.0027
Epoch 40/100
38/38 [==============================] - 5s 126ms/step - loss: 0.0028
Epoch 41/100
38/38 [==============================] - 4s 103ms/step - loss: 0.0025
Epoch 42/100
38/38 [==============================] - 4s 111ms/step - loss: 0.0024
Epoch 43/100
38/38 [==============================] - 5s 124ms/step - loss: 0.0024
Epoch 44/100
38/38 [==============================] - 4s 106ms/step - loss: 0.0028
Epoch 45/100
38/38 [==============================] - 4s 117ms/step - loss: 0.0025
Epoch 46/100
38/38 [==============================] - 4s 108ms/step - loss: 0.0023
Epoch 47/100
38/38 [==============================] - 5s 129ms/step - loss: 0.0024
Epoch 48/100
38/38 [==============================] - 4s 107ms/step - loss: 0.0022
Epoch 49/100
38/38 [==============================] - 5s 126ms/step - loss: 0.0022
Epoch 50/100
38/38 [==============================] - 5s 120ms/step - loss: 0.0023
Epoch 51/100
38/38 [==============================] - 4s 112ms/step - loss: 0.0025
Epoch 52/100
38/38 [==============================] - 4s 115ms/step - loss: 0.0022
Epoch 53/100
38/38 [==============================] - 4s 114ms/step - loss: 0.0024
Epoch 54/100
38/38 [==============================] - 5s 121ms/step - loss: 0.0020
Epoch 55/100
38/38 [==============================] - 4s 112ms/step - loss: 0.0022
Epoch 56/100
38/38 [==============================] - 5s 124ms/step - loss: 0.0025
Epoch 57/100
38/38 [==============================] - 4s 102ms/step - loss: 0.0021
Epoch 58/100
38/38 [==============================] - 5s 131ms/step - loss: 0.0021
Epoch 59/100
38/38 [==============================] - 4s 110ms/step - loss: 0.0023
Epoch 60/100
38/38 [==============================] - 5s 121ms/step - loss: 0.0019
Epoch 61/100
38/38 [==============================] - 4s 110ms/step - loss: 0.0018
Epoch 62/100
38/38 [==============================] - 4s 116ms/step - loss: 0.0021
Epoch 63/100
38/38 [==============================] - 4s 113ms/step - loss: 0.0020
Epoch 64/100
38/38 [==============================] - 4s 104ms/step - loss: 0.0019
Epoch 65/100
38/38 [==============================] - 4s 113ms/step - loss: 0.0019
Epoch 66/100
38/38 [==============================] - 4s 116ms/step - loss: 0.0018
Epoch 67/100
38/38 [==============================] - 5s 121ms/step - loss: 0.0019
Epoch 68/100
38/38 [==============================] - 5s 121ms/step - loss: 0.0018
Epoch 69/100
38/38 [==============================] - 4s 116ms/step - loss: 0.0017
Epoch 70/100
38/38 [==============================] - 4s 109ms/step - loss: 0.0021
Epoch 71/100
38/38 [==============================] - 4s 109ms/step - loss: 0.0019
Epoch 72/100
38/38 [==============================] - 4s 111ms/step - loss: 0.0017
Epoch 73/100
38/38 [==============================] - 4s 112ms/step - loss: 0.0018
Epoch 74/100
38/38 [==============================] - 4s 101ms/step - loss: 0.0018
Epoch 75/100
38/38 [==============================] - 5s 124ms/step - loss: 0.0016
```

```
Epoch 76/100
38/38 [==============================] - 4s 113ms/step - loss: 0.0017
Epoch 77/100
38/38 [==============================] - 4s 118ms/step - loss: 0.0019
Epoch 78/100
38/38 [==============================] - 5s 121ms/step - loss: 0.0016
Epoch 79/100
38/38 [==============================] - 4s 114ms/step - loss: 0.0015
Epoch 80/100
38/38 [==============================] - 5s 123ms/step - loss: 0.0016
Epoch 81/100
38/38 [==============================] - 5s 126ms/step - loss: 0.0017
Epoch 82/100
38/38 [==============================] - 4s 98ms/step - loss: 0.0018
Epoch 83/100
38/38 [==============================] - 5s 125ms/step - loss: 0.0017
Epoch 84/100
38/38 [==============================] - 4s 103ms/step - loss: 0.0016
Epoch 85/100
38/38 [==============================] - 5s 121ms/step - loss: 0.0015
Epoch 86/100
38/38 [==============================] - 4s 116ms/step - loss: 0.0015
Epoch 87/100
38/38 [==============================] - 4s 98ms/step - loss: 0.0018
Epoch 88/100
38/38 [==============================] - 4s 109ms/step - loss: 0.0015
Epoch 89/100
38/38 [==============================] - 4s 117ms/step - loss: 0.0016
Epoch 90/100
38/38 [==============================] - 5s 121ms/step - loss: 0.0015
Epoch 91/100
38/38 [==============================] - 5s 123ms/step - loss: 0.0015
Epoch 92/100
38/38 [==============================] - 5s 121ms/step - loss: 0.0015
Epoch 93/100
38/38 [==============================] - 4s 119ms/step - loss: 0.0015
Epoch 94/100
38/38 [==============================] - 4s 110ms/step - loss: 0.0015
Epoch 95/100
38/38 [==============================] - 4s 118ms/step - loss: 0.0014
Epoch 96/100
38/38 [==============================] - 4s 115ms/step - loss: 0.0014
Epoch 97/100
38/38 [==============================] - 4s 114ms/step - loss: 0.0013
Epoch 98/100
38/38 [==============================] - 5s 128ms/step - loss: 0.0014
Epoch 99/100
38/38 [==============================] - 4s 115ms/step - loss: 0.0015
Epoch 100/100
38/38 [==============================] - 5s 126ms/step - loss: 0.0014
```

Out[12]:

```
<keras.callbacks.History at 0x2197ed84c40>
```

In [13]:

```python
dataset_test = pd.read_csv('Google_Stock_Price_Test.csv')
real_stock_price = dataset_test.iloc[:, 1:2].values

# Getting the predicted stock price of 2017
dataset_total = pd.concat((dataset_train['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)
print(predicted_stock_price)
```

```
1/1 [==============================] - 1s 1s/step
[[778.2358 ]
 [775.4508 ]
 [775.7869 ]
 [776.96094]
 [779.9706 ]
 [785.5475 ]
 [790.5848 ]
 [792.6368 ]
 [793.3651 ]
 [793.78827]
 [794.0905 ]
 [794.06726]
 [793.8742 ]
 [794.28375]
 [795.0378 ]
 [799.56537]
 [806.1648 ]
 [813.13776]
 [816.8018 ]
 [812.707  ]]
```

In [14]:

```
# Visualising the results
plt.plot(real_stock_price, color = 'red', label = 'Real Google Stock Price')
plt.plot(predicted_stock_price, color = 'blue', label = 'Predicted Google Stock Price')
plt.title('Google Stock Price Prediction')
plt.xlabel('Time')
plt.ylabel('Google Stock Price')
plt.legend()
plt.show()
```