In [2]:
```
!python --version

```

```
Python 3.8.16
```

In [4]:
```
!pip install multitasking==0.0.9 --force-reinstall
!pip install yfinance==0.2.28 --force-reinstall --user
```

```
8)
  Obtaining dependency information for charset_normalizer<4,>=2 from https://files.pythonhosted.org/packages/7c/02/1c82646582ccf2c757fa6af69b1a3ea88744b8d2b4ab93b7686b2533e023/charset_normalizer-3.4.2-cp38-cp38-win_amd64.whl.metadata (https://files.pythonhosted.org/packages/7c/02/1c82646582ccf2c757fa6af69b1a3ea88744b8d2b4ab93b7686b2533e023/charset_normalizer-3.4.2-cp38-cp38-win_amd64.whl.metadata)
  Using cached charset_normalizer-3.4.2-cp38-cp38-win_amd64.whl.metadata (36 kB)
Collecting idna<4,>=2.5 (from requests>=2.31->yfinance==0.2.28)
  Obtaining dependency information for idna<4,>=2.5 from https://files.pythonhosted.org/packages/76/c6/c88e154df9c4e1a2a66ccf0005a88dfb2650c1dffb6f5ce603dfbd452ce3/idna-3.10-py3-none-any.whl.metadata (https://files.pythonhosted.org/packages/76/c6/c88e154df9c4e1a2a66ccf0005a88dfb2650c1dffb6f5ce603dfbd452ce3/idna-3.10-py3-none-any.whl.metadata)
  Using cached idna-3.10-py3-none-any.whl.metadata (10 kB)
Collecting urllib3<3,>=1.21.1 (from requests>=2.31->yfinance==0.2.28)
  Obtaining dependency information for urllib3<3,>=1.21.1 from https://files.pythonhosted.org/packages/ce/d9/5f4c13cecde62396b0d3fe530a50ccea91e7dfc1ccf0e09c228841bb5ba8/urllib3-2.2.3-py3-none-any.whl.metadata (https://
```

In [5]:

```
!pip install pandas_datareader
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import pandas_datareader as data
```

```
Requirement already satisfied: pandas_datareader in c:\users\rahul\anaconda3
\lib\site-packages (0.10.0)
Requirement already satisfied: lxml in c:\users\rahul\appdata\roaming\python
\python38\site-packages (from pandas_datareader) (6.0.0)
Requirement already satisfied: pandas>=0.23 in c:\users\rahul\appdata\roamin
g\python\python38\site-packages (from pandas_datareader) (2.0.3)
Requirement already satisfied: requests>=2.19.0 in c:\users\rahul\appdata\ro
aming\python\python38\site-packages (from pandas_datareader) (2.32.4)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\rahul\appd
ata\roaming\python\python38\site-packages (from pandas>=0.23->pandas_datarea
der) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\rahul\appdata\roamin
g\python\python38\site-packages (from pandas>=0.23->pandas_datareader) (202
5.2)
Requirement already satisfied: tzdata>=2022.1 in c:\users\rahul\appdata\roam
ing\python\python38\site-packages (from pandas>=0.23->pandas_datareader) (20
25.2)
Requirement already satisfied: numpy>=1.20.3 in c:\users\rahul\appdata\roami
ng\python\python38\site-packages (from pandas>=0.23->pandas_datareader) (1.2
4.4)
Requirement already satisfied: charset_normalizer<4,>=2 in c:\users\rahul\ap
pdata\roaming\python\python38\site-packages (from requests>=2.19.0->pandas_d
atareader) (3.4.2)
Requirement already satisfied: idna<4,>=2.5 in c:\users\rahul\appdata\roamin
g\python\python38\site-packages (from requests>=2.19.0->pandas_datareader)
(3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\rahul\appdata
\roaming\python\python38\site-packages (from requests>=2.19.0->pandas_datare
ader) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\rahul\appdata
\roaming\python\python38\site-packages (from requests>=2.19.0->pandas_datare
ader) (2025.7.14)
Requirement already satisfied: six>=1.5 in c:\users\rahul\appdata\roaming\py
thon\python38\site-packages (from python-dateutil>=2.8.2->pandas>=0.23->pand
as_datareader) (1.17.0)
```

In [6]:

```
import yfinance as yf

# Example usage
df = yf.download("AAPL", start="2010-01-01", end="2019-12-31")
df.head()
```

```
C:\Users\Rahul\AppData\Local\Temp\ipykernel_26596\1298867081.py:4: FutureWar
ning: YF.download() has changed argument auto_adjust default to True
  df = yf.download("AAPL", start="2010-01-01", end="2019-12-31")
[*********************100%***********************]  1 of 1 completed
```

Out[6]:

Price	Close	High	Low	Open	Volume
Ticker	AAPL	AAPL	AAPL	AAPL	AAPL
Date					
2010-01-04	6.431898	6.446625	6.382910	6.414467	493729600
2010-01-05	6.443017	6.479383	6.409056	6.449630	601904800
2010-01-06	6.340531	6.468562	6.333919	6.443016	552160000
2010-01-07	6.328809	6.371487	6.282827	6.363973	477131200
2010-01-08	6.370885	6.371487	6.283128	6.320394	447610800

In [7]:

```
df = df.reset_index()
df
```

Out[7]:

Price	Date	Close	High	Low	Open	Volume
Ticker		AAPL	AAPL	AAPL	AAPL	AAPL
0	2010-01-04	6.431898	6.446625	6.382910	6.414467	493729600
1	2010-01-05	6.443017	6.479383	6.409056	6.449630	601904800
2	2010-01-06	6.340531	6.468562	6.333919	6.443016	552160000
3	2010-01-07	6.328809	6.371487	6.282827	6.363973	477131200
4	2010-01-08	6.370885	6.371487	6.283128	6.320394	447610800
...	...	...	...	...	...	...
2510	2019-12-23	68.667610	68.728057	67.789921	67.828608	98572000
2511	2019-12-24	68.732887	68.882802	68.406480	68.834441	48478800
2512	2019-12-26	70.096565	70.113492	68.836855	68.865869	93121200
2513	2019-12-27	70.069977	71.078233	69.663776	70.389138	146266000
2514	2019-12-30	70.485847	70.768742	68.962590	69.987766	144114400

2515 rows × 6 columns

In [9]:
```
df.columns
```

Out[9]:
```
MultiIndex([(  'Date',     ''),
            ( 'Close', 'AAPL'),
            (  'High', 'AAPL'),
            (   'Low', 'AAPL'),
            (  'Open', 'AAPL'),
            ('Volume', 'AAPL')],
           names=['Price', 'Ticker'])
```

In [10]:
```
df=df.drop(['Date'], axis=1)
df.head()
```

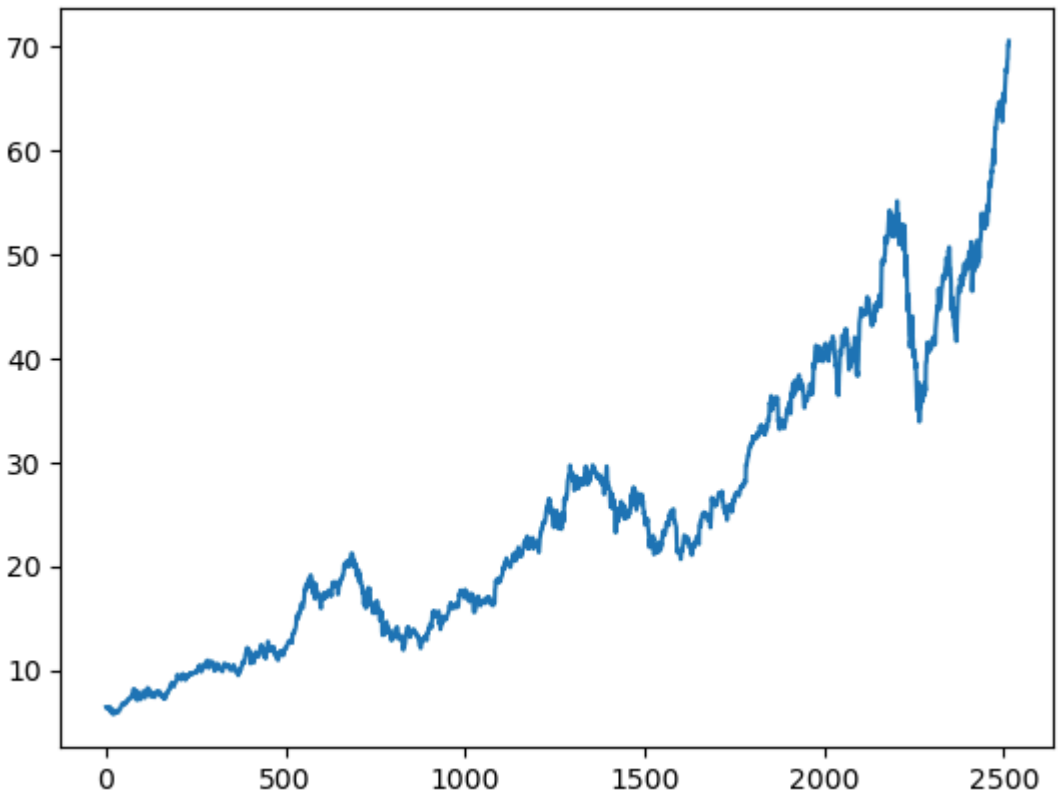
```
C:\Users\Rahul\AppData\Local\Temp\ipykernel_26596\2547152359.py:1: Performan
ceWarning: dropping on a non-lexsorted multi-index without a level parameter
may impact performance.
  df=df.drop(['Date'], axis=1)
```

Out[10]:

Price	Close	High	Low	Open	Volume
Ticker	AAPL	AAPL	AAPL	AAPL	AAPL
0	6.431898	6.446625	6.382910	6.414467	493729600
1	6.443017	6.479383	6.409056	6.449630	601904800
2	6.340531	6.468562	6.333919	6.443016	552160000
3	6.328809	6.371487	6.282827	6.363973	477131200
4	6.370885	6.371487	6.283128	6.320394	447610800

In [12]:
```
plt.plot(df.Close)
```

Out[12]:
```
[<matplotlib.lines.Line2D at 0x24655c6c4f0>]
```



In [13]:
```
1 df
```

Out[13]:

Price	Close	High	Low	Open	Volume
Ticker	AAPL	AAPL	AAPL	AAPL	AAPL
0	6.431898	6.446625	6.382910	6.414467	493729600
1	6.443017	6.479383	6.409056	6.449630	601904800
2	6.340531	6.468562	6.333919	6.443016	552160000
3	6.328809	6.371487	6.282827	6.363973	477131200
4	6.370885	6.371487	6.283128	6.320394	447610800
...	...	...	...	...	...
2510	68.667610	68.728057	67.789921	67.828608	98572000
2511	68.732887	68.882802	68.406480	68.834441	48478800
2512	70.096565	70.113492	68.836855	68.865869	93121200
2513	70.069977	71.078233	69.663776	70.389138	146266000
2514	70.485847	70.768742	68.962590	69.987766	144114400

2515 rows × 5 columns

In [14]:
```
1 df['Close'].describe()
```

Out[14]:

Ticker	AAPL
count	2515.000000
mean	25.142499
std	13.904044
min	5.771907
25%	13.809111
50%	22.281395
75%	35.221708
max	70.485847

In [15]:
```
ma100 =df.Close.rolling(100).mean()
ma100
```

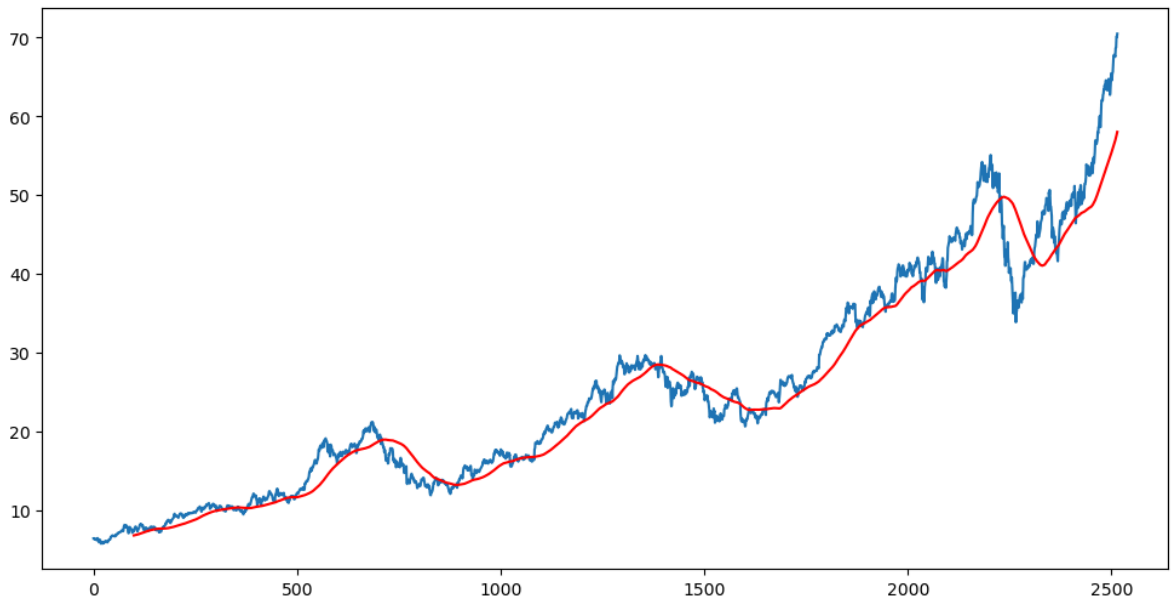
Out[15]:

Ticker	AAPL
0	NaN
1	NaN
2	NaN
3	NaN
4	NaN
...	...
2510	57.137912
2511	57.335284
2512	57.571942
2513	57.799544
2514	58.026405

2515 rows × 1 columns

In [17]:
```
plt.figure(figsize = (12,6))
plt.plot(df.Close)
plt.plot(ma100, 'r')
```

Out[17]:
```
[<matplotlib.lines.Line2D at 0x24657e77ac0>]
```



In [18]:
```
df.shape
```

Out[18]:
```
(2515, 5)
```

In [22]:
```
ma200 =df.Close.rolling(200).mean()
ma200
```

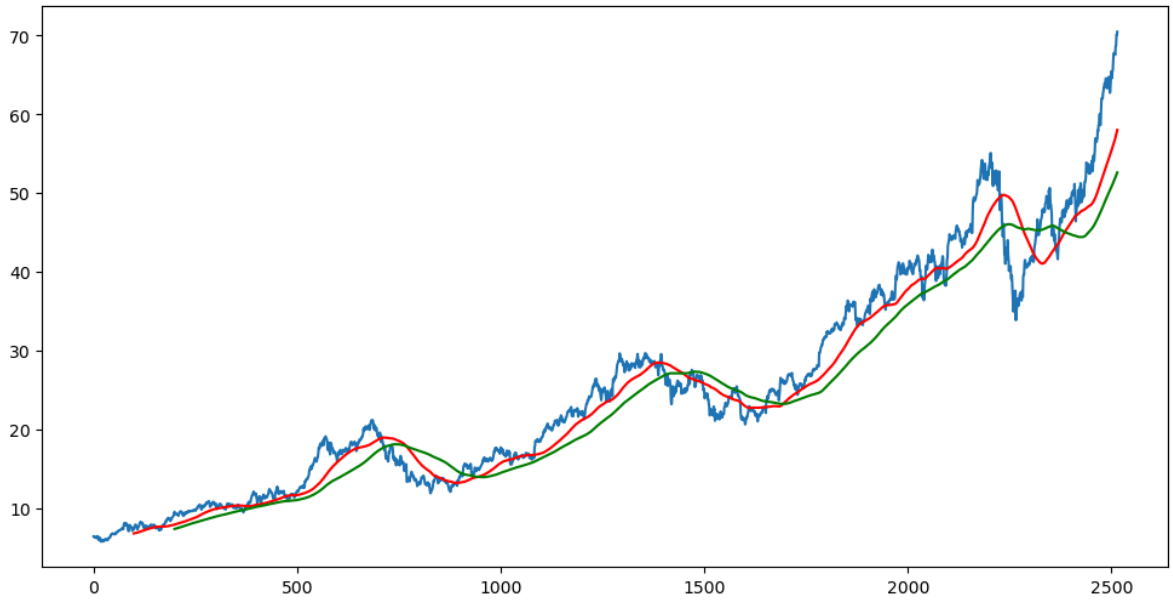
Out[22]:

Ticker	AAPL
0	NaN
1	NaN
2	NaN
3	NaN
4	NaN
...	...
2510	52.096150
2511	52.223419
2512	52.356549
2513	52.487130
2514	52.616932

2515 rows × 1 columns

In [23]:
```
plt.figure(figsize = (12,6))
plt.plot(df.Close)
plt.plot(ma100, 'r')
plt.plot(ma200, 'g')
```

Out[23]: [<matplotlib.lines.Line2D at 0x24657ee21c0>]



In [21]:
```
#splitting data into training and testing

data_training= pd.DataFrame(df['Close'][0:int(len(df)*0.70)])
data_testing= pd.DataFrame(df['Close'][int(len(df)*0.70): int(len(df))])

print(data_training.shape)
print(data_testing.shape)

```

```
(1760, 1)
(755, 1)
```

In [24]:
```
data_training.head()
```

Out[24]:

Ticker	AAPL
0	6.431898
1	6.443017
2	6.340531
3	6.328809
4	6.370885

In [25]:
```
data_testing.head()
```

Out[25]:

Ticker	AAPL
1760	26.961208
1761	26.751022
1762	26.827248
1763	26.797222
1764	26.933495

In [26]:
```
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler(feature_range=(0,1))
```

In [27]:
```
data_training_array= scaler.fit_transform(data_training)
data_training_array
```

Out[27]:
```
array([[0.02760007],
       [0.02806507],
       [0.02377919],
       ...,
       [0.88408442],
       [0.89123176],
       [0.8864025 ]])
```

In [34]:
```
print("101st and 102nd row of data_training_array:")
print(data_training_array[100])
print(data_training_array[101])
```

```
101st and 102nd row of data_training_array:
[0.07704382]
[0.08148037]
```

In [35]:
```
x_train = []
y_train = []

for i in range(100, data_training_array.shape[0]):
    x_train.append(data_training_array[i-100: i ])
    y_train.append(data_training_array[i,0])


#x_train
#y_train
x_train, y_train = np.array(x_train), np.array(y_train)
```

In [36]:
```
x_train.shape
```

Out[36]:
```
(1660, 100, 1)
```

In [45]:
```
from keras.models import Sequential
from keras.layers import LSTM, Dropout, Dense

model = Sequential()

model.add(LSTM(units=50, activation='relu', return_sequences=True, input_
model.add(Dropout(0.2))

model.add(LSTM(units=60, activation='relu', return_sequences=True))
model.add(Dropout(0.3))

model.add(LSTM(units=80, activation='relu', return_sequences=True))
model.add(Dropout(0.4))

model.add(LSTM(units=120, activation='relu'))
model.add(Dropout(0.5))

model.add(Dense(units=1))
```

In [46]:
```
print(x_train.shape)
```

```
(1660, 100, 1)
```

In [47]:

```
1 model.summary()
```

```
Model: "sequential_12"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_12 (LSTM)              (None, 100, 50)           10400

 dropout_12 (Dropout)        (None, 100, 50)           0

 lstm_13 (LSTM)              (None, 100, 60)           26640

 dropout_13 (Dropout)        (None, 100, 60)           0

 lstm_14 (LSTM)              (None, 100, 80)           45120

 dropout_14 (Dropout)        (None, 100, 80)           0

 lstm_15 (LSTM)              (None, 120)               96480

 dropout_15 (Dropout)        (None, 120)               0

 dense_3 (Dense)             (None, 1)                 121

=================================================================
Total params: 178761 (698.29 KB)
Trainable params: 178761 (698.29 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

In [48]:

```
model.compile(optimizer='adam', loss= 'mean_squared_error')
model.fit(x_train, y_train, epochs=50)
```

```
Epoch 1/50
52/52 [==============================] - 39s 496ms/step - loss: 0.0667
Epoch 2/50
52/52 [==============================] - 25s 488ms/step - loss: 0.0134
Epoch 3/50
52/52 [==============================] - 18s 352ms/step - loss: 0.0107
Epoch 4/50
52/52 [==============================] - 17s 318ms/step - loss: 0.0106
Epoch 5/50
52/52 [==============================] - 23s 455ms/step - loss: 0.0091
Epoch 6/50
52/52 [==============================] - 24s 467ms/step - loss: 0.0090
Epoch 7/50
52/52 [==============================] - 25s 471ms/step - loss: 0.0094
Epoch 8/50
52/52 [==============================] - 25s 483ms/step - loss: 0.0077
Epoch 9/50
52/52 [==============================] - 25s 482ms/step - loss: 0.0082
Epoch 10/50
52/52 [==============================] - 27s 514ms/step - loss: 0.0077
Epoch 11/50
52/52 [==============================] - 26s 499ms/step - loss: 0.0072
Epoch 12/50
52/52 [==============================] - 26s 498ms/step - loss: 0.0067
Epoch 13/50
52/52 [==============================] - 26s 509ms/step - loss: 0.0065
Epoch 14/50
52/52 [==============================] - 27s 521ms/step - loss: 0.0074
Epoch 15/50
52/52 [==============================] - 27s 522ms/step - loss: 0.0072
Epoch 16/50
52/52 [==============================] - 27s 515ms/step - loss: 0.0068
Epoch 17/50
52/52 [==============================] - 26s 508ms/step - loss: 0.0063
Epoch 18/50
52/52 [==============================] - 27s 519ms/step - loss: 0.0062
Epoch 19/50
52/52 [==============================] - 27s 522ms/step - loss: 0.0059
Epoch 20/50
52/52 [==============================] - 27s 510ms/step - loss: 0.0058
Epoch 21/50
52/52 [==============================] - 26s 500ms/step - loss: 0.0054
Epoch 22/50
52/52 [==============================] - 27s 513ms/step - loss: 0.0067
Epoch 23/50
52/52 [==============================] - 26s 508ms/step - loss: 0.0056
Epoch 24/50
52/52 [==============================] - 27s 510ms/step - loss: 0.0062
Epoch 25/50
52/52 [==============================] - 27s 512ms/step - loss: 0.0058
Epoch 26/50
52/52 [==============================] - 27s 511ms/step - loss: 0.0055
Epoch 27/50
52/52 [==============================] - 27s 520ms/step - loss: 0.0051
Epoch 28/50
52/52 [==============================] - 26s 506ms/step - loss: 0.0053
Epoch 29/50
52/52 [==============================] - 27s 510ms/step - loss: 0.0050
Epoch 30/50
52/52 [==============================] - 24s 455ms/step - loss: 26.5089
Epoch 31/50
```

```
52/52 [==============================] - 27s 510ms/step - loss: 0.0073
Epoch 32/50
52/52 [==============================] - 27s 516ms/step - loss: 0.0056
Epoch 33/50
52/52 [==============================] - 27s 511ms/step - loss: 0.0054
Epoch 34/50
52/52 [==============================] - 26s 497ms/step - loss: 0.0052
Epoch 35/50
52/52 [==============================] - 25s 481ms/step - loss: 0.0052
Epoch 36/50
52/52 [==============================] - 21s 402ms/step - loss: 0.0047
Epoch 37/50
52/52 [==============================] - 24s 456ms/step - loss: 0.0050
Epoch 38/50
52/52 [==============================] - 25s 490ms/step - loss: 0.0047
Epoch 39/50
52/52 [==============================] - 23s 447ms/step - loss: 0.0048
Epoch 40/50
52/52 [==============================] - 26s 508ms/step - loss: 0.0043
Epoch 41/50
52/52 [==============================] - 26s 508ms/step - loss: 0.0049
Epoch 42/50
52/52 [==============================] - 26s 504ms/step - loss: 0.0040
Epoch 43/50
52/52 [==============================] - 27s 518ms/step - loss: 0.0045
Epoch 44/50
52/52 [==============================] - 26s 495ms/step - loss: 0.0043
Epoch 45/50
52/52 [==============================] - 24s 469ms/step - loss: 0.0046
Epoch 46/50
52/52 [==============================] - 27s 516ms/step - loss: 0.0048
Epoch 47/50
52/52 [==============================] - 24s 456ms/step - loss: 0.0044
Epoch 48/50
52/52 [==============================] - 26s 504ms/step - loss: 0.0044
Epoch 49/50
52/52 [==============================] - 25s 482ms/step - loss: 0.0040
Epoch 50/50
52/52 [==============================] - 27s 517ms/step - loss: 0.0043
```

Out[48]:
```
<keras.src.callbacks.History at 0x24609b907f0>
```

In [49]:
```
1 model.save('keras_model.h5')
```

```
C:\Users\Rahul\Anaconda3\lib\site-packages\keras\src\engine\training.py:300
0: UserWarning: You are saving your model as an HDF5 file via `model.save()
`. This file format is considered legacy. We recommend using instead the nat
ive Keras format, e.g. `model.save('my_model.keras')`.
  saving_api.save_model(
```

In [50]:
```
1 data_testing.head()
2
```

Out[50]:

Ticker	AAPL
1760	26.961208
1761	26.751022
1762	26.827248
1763	26.797222
1764	26.933495

In [51]:
```
1 data_training.tail(100)
```

Out[51]:

Ticker	AAPL
1660	24.902437
1661	25.003546
1662	24.817421
1663	24.801332
1664	24.858780
...	...
1755	27.037430
1756	26.859585
1757	26.912708
1758	27.083620
1759	26.968140

100 rows × 1 columns

In [52]:
```
1 past_100_days= data_training.tail(100)
```

In [54]:
```
final_df=past_100_days.append(data_testing, ignore_index=True)
final_df
```

```
C:\Users\Rahul\AppData\Local\Temp\ipykernel_26596\3052075165.py:1: FutureWar
ning: The frame.append method is deprecated and will be removed from pandas
in a future version. Use pandas.concat instead.
  final_df=past_100_days.append(data_testing, ignore_index=True)
```

Out[54]:

Ticker	AAPL
0	24.902437
1	25.003546
2	24.817421
3	24.801332
4	24.858780
...	...
850	68.667610
851	68.732887
852	70.096565
853	70.069977
854	70.485847

855 rows × 1 columns

In [57]:
```
input_data=scaler.fit_transform(final_df)
input_data
```

```
       [0.83426962],
       [0.84610412],
       [0.86594788],
       [0.89245846],
       [0.87287308],
       [0.88093484],
       [0.89276894],
       [0.89633484],
       [0.91540379],
       [0.93974415],
       [0.94258604],
       [0.93912401],
       [0.94057072],
       [0.93757343],
       [0.96113841],
       [0.96253359],
       [0.99167978],
       [0.9911115 ],
       [1.        ]])
```

In [58]:
```
input_data.shape
```

Out[58]: (855, 1)

In [59]:
```
x_test =[]
y_test= []

for i in range(100, input_data.shape[0]):
    x_test.append(input_data[i-100: i])
    y_test.append(input_data[i, 0])
```

In [62]:
```
#x_test
#y_test

x_test, y_test = np.array(x_test), np.array(y_test)
print(x_test.shape)
print(y_test.shape)
```

```
(755, 100, 1)
(755,)
```

In [63]:
```
# Making predictions

y_predicted= model.predict(x_test)
```

```
24/24 [==============================] - 6s 173ms/step
```

In [64]:
```
y_predicted.shape
```

Out[64]:
```
(755, 1)
```

In [65]:
```
y_predicted
```

Out[65]:
```
array([[0.07691284],
       [0.077049  ],
       [0.07718353],
       [0.07731378],
       [0.07743762],
       [0.07755409],
       [0.07766412],
       [0.07777026],
       [0.07787526],
       [0.07798339],
       [0.07809825],
       [0.07822181],
       [0.07835655],
       [0.07850416],
       [0.07866482],
       [0.07883786],
       [0.07902196],
       [0.07921503],
       [0.07941839],
```

In [66]:
```
scaler.scale_
```

Out[66]:
```
array([0.02137322])
```

In [67]:
```
scale_factor = 1/0.02137322
y_predicted=y_predicted * scale_factor
y_test = y_test * scale_factor
```

In [68]:
```
y_predicted
```

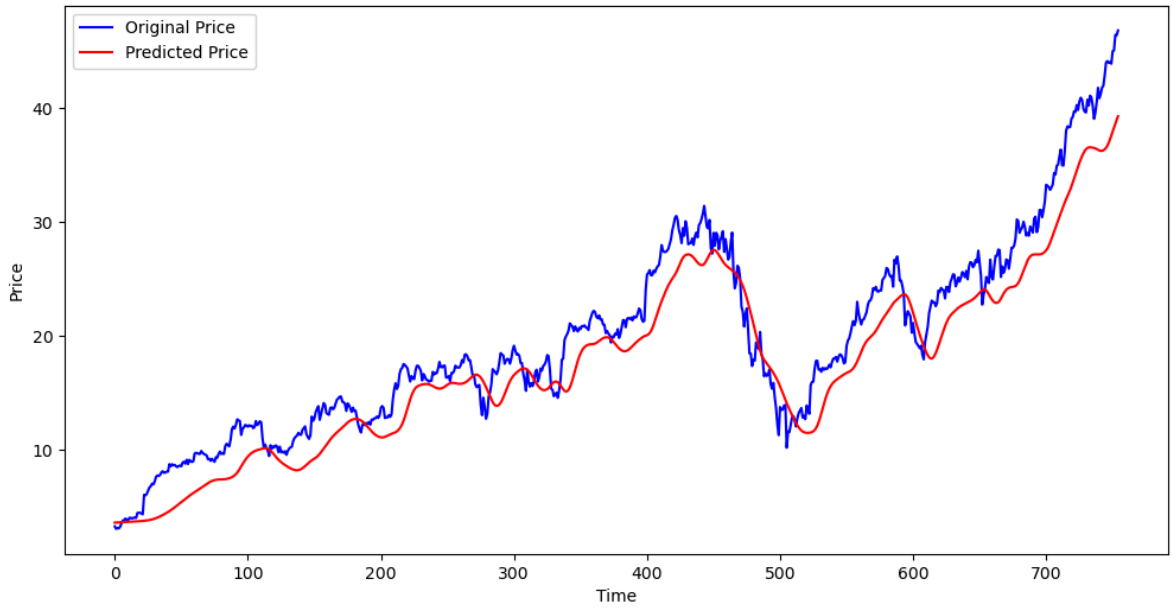
Out[68]:
```
array([[ 3.5985608],
       [ 3.6049318],
       [ 3.611226 ],
       [ 3.61732  ],
       [ 3.6231146],
       [ 3.6285636],
       [ 3.6337118],
       [ 3.6386778],
       [ 3.6435902],
       [ 3.6486497],
       [ 3.6540236],
       [ 3.6598048],
       [ 3.6661086],
       [ 3.673015 ],
       [ 3.6805322],
       [ 3.688628 ],
       [ 3.6972418],
       [ 3.7062752],
       [ 3.7157896],
```

In [69]:
```
y_test
```

```
16.4707353 , 16.58268525, 17.02337061, 15.72036893, 15.35353612,
15.86091385, 14.62700779, 13.65988264, 12.20680692, 11.2777906 ,
13.74086105, 13.4978953 , 13.51693446, 13.87664142, 13.91951099,
10.17251235, 11.61842788, 11.5398184 , 12.21156385, 12.82137385,
12.93810743, 12.57841192, 12.03291772, 12.76419916, 13.20966053,
13.42881875, 13.65749464, 12.8189973 , 12.96667952, 12.67607582,
13.88140979, 13.53362758, 13.14772509, 15.66559368, 15.94905626,
15.96810305, 17.09483517, 17.79278377, 17.80707745, 17.02099406,
17.068838  , 16.83439428, 17.18366804, 17.01381479, 17.16213407,
17.07122982, 17.19322768, 17.45639225, 17.22433272, 17.68126489,
17.9826947 , 18.00661668, 18.13580143, 17.72433283, 18.15972721,
18.37023751, 18.2936956 , 18.05206499, 17.56883048, 17.666914  ,
19.09990533, 19.58075566, 19.77213906, 20.25537739, 20.82714722,
21.28167995, 20.92523074, 21.31518062, 22.97304079, 22.00654506,
21.45392119, 20.98742558, 21.38932691, 21.44913756, 21.74339574,
22.05200482, 22.71705929, 23.03522799, 23.11657262, 23.42996914,
24.17157696, 24.02804132, 24.2959819 , 23.89645712, 23.87731497,
23.96342796, 23.96823448, 24.8964459 , 25.07107896, 25.23136494,
25.93709172, 25.86054217, 25.41078545, 25.17634937, 25.25049947,
24.30793335, 26.66434484, 26.33660505, 26.95860683, 26.17633433,
```

In [72]:

```
plt.figure(figsize=(12,6))
plt.plot(y_test, 'b', label= 'Original Price')
plt.plot(y_predicted, 'r', label = 'Predicted Price')
plt.xlabel('Time')
plt.ylabel('Price')
plt.legend()
plt.show()
```



In [ ]:

```
1
```