
## >Beginner Level Task...

## Task-2 Stock Market Prediction And Forecasting Using Stacked LSTM :

Dataset: https://raw.githubusercontent.com/mwitiderrick/stockprice/master/NSE-TATAGLOBAL.csv

Name: Manish Singh

### 1. Importing Required Libraries and Packages

In [3]:
```
import numpy as np
import pandas as pd
import datetime
import math
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense , LSTM
from sklearn.metrics import mean_squared_error
print(" All required packages included successfully!")
```
```
 All required packages included successfully!
```

### 2. Importing the Dataset

In [5]:
```
Dataset_link='https://raw.githubusercontent.com/mwitiderrick/stockprice/master/NSE-TATAGLOBAL.csv'
```

In [6]:
```
df= pd.read_csv(Dataset_link, parse_dates=True,)
df.reset_index()
df.head(10)
```

Out[6]:

| | Date | Open | High | Low | Last | Close | Total Trade Quantity | Turnover (Lacs) |
|---|---|---|---|---|---|---|---|---|
| 0 | 2018-09-28 | 234.05 | 235.95 | 230.20 | 233.50 | 233.75 | 3069914 | 7162.35 |
| 1 | 2018-09-27 | 234.55 | 236.80 | 231.10 | 233.80 | 233.25 | 5082859 | 11859.95 |
| 2 | 2018-09-26 | 240.00 | 240.00 | 232.50 | 235.00 | 234.25 | 2240909 | 5248.60 |
| 3 | 2018-09-25 | 233.30 | 236.75 | 232.00 | 236.25 | 236.10 | 2349368 | 5503.90 |
| 4 | 2018-09-24 | 233.55 | 239.20 | 230.75 | 234.00 | 233.30 | 3423509 | 7999.55 |
| 5 | 2018-09-21 | 235.00 | 237.00 | 227.95 | 233.75 | 234.60 | 5395319 | 12589.59 |
| 6 | 2018-09-19 | 235.95 | 237.20 | 233.45 | 234.60 | 234.90 | 1362058 | 3202.78 |
| 7 | 2018-09-18 | 237.90 | 239.25 | 233.50 | 235.50 | 235.05 | 2614794 | 6163.70 |
| 8 | 2018-09-17 | 233.15 | 238.00 | 230.25 | 236.40 | 236.60 | 3170894 | 7445.41 |
| 9 | 2018-09-14 | 223.45 | 236.70 | 223.30 | 234.00 | 233.95 | 6377909 | 14784.50 |

### 3. Performing EDA

In [7]:
```
df.sample(10)
```

Out[7]:

| | Date | Open | High | Low | Last | Close | Total Trade Quantity | Turnover (Lacs) |
|---|---|---|---|---|---|---|---|---|
| 95 | 2018-05-14 | 280.00 | 281.35 | 251.85 | 257.00 | 255.90 | 16822847 | 43913.37 |
| 1333 | 2013-05-10 | 148.05 | 149.20 | 145.10 | 146.40 | 146.15 | 1749193 | 2566.83 |
| 1275 | 2013-07-30 | 160.45 | 163.60 | 155.30 | 159.00 | 158.85 | 7626988 | 12233.65 |
| 580 | 2016-05-30 | 118.00 | 119.65 | 117.30 | 117.90 | 117.95 | 692008 | 819.71 |
| 1874 | 2011-03-09 | 94.75 | 98.20 | 94.05 | 96.20 | 96.35 | 3190205 | 3069.91 |
| 159 | 2018-02-06 | 259.00 | 270.30 | 253.25 | 266.00 | 266.70 | 7044561 | 18431.55 |
| 1672 | 2012-01-02 | 90.95 | 91.90 | 89.10 | 90.75 | 90.95 | 694783 | 632.80 |
| 719 | 2015-11-03 | 133.30 | 134.15 | 130.80 | 131.50 | 131.75 | 519168 | 686.80 |
| 1079 | 2014-05-19 | 146.00 | 151.60 | 145.25 | 150.60 | 150.70 | 4970218 | 7374.52 |
| 185 | 2017-12-29 | 308.05 | 318.00 | 306.35 | 316.05 | 316.40 | 6874520 | 21586.26 |

In [8]:
```
df.columns
```

Out[8]:
```
Index(['Date', 'Open', 'High', 'Low', 'Last', 'Close', 'Total Trade Quantity',
       'Turnover (Lacs)'],
      dtype='object')
```

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

Out[9]:
```
(2035, 8)
```

In [10]:
```
df.info()
```
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2035 entries, 0 to 2034
Data columns (total 8 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Date                  2035 non-null   object
 1   Open                  2035 non-null   float64
 2   High                  2035 non-null   float64
 3   Low                   2035 non-null   float64
 4   Last                  2035 non-null   float64
 5   Close                 2035 non-null   float64
 6   Total Trade Quantity  2035 non-null   int64
 7   Turnover (Lacs)       2035 non-null   float64
dtypes: float64(6), int64(1), object(1)
memory usage: 127.3+ KB
```

In [11]:
```
df.isnull().sum()
```

Out[11]:
```
Date                    0
Open                    0
High                    0
Low                     0
Last                    0
Close                   0
Total Trade Quantity    0
Turnover (Lacs)         0
dtype: int64
```

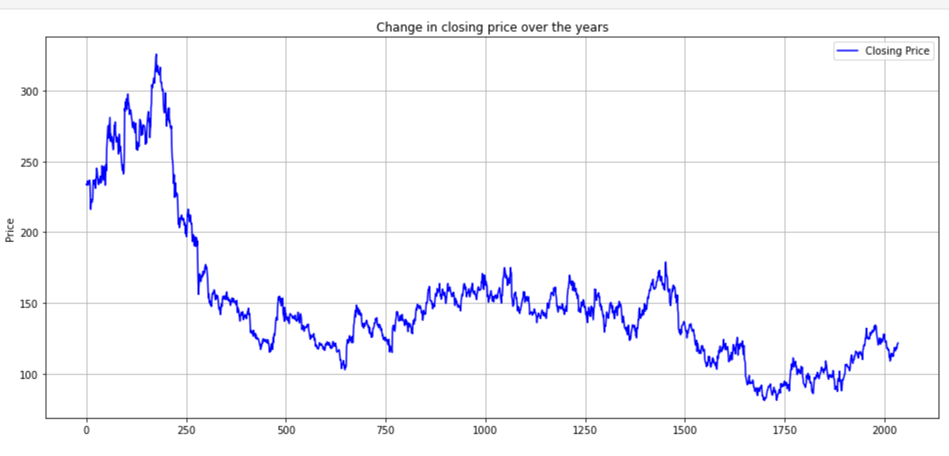
In [12]:
```
df.describe()
```

Out[12]:

| | Open | High | Low | Last | Close | Total Trade Quantity | Turnover (Lacs) |
|---|---|---|---|---|---|---|---|
| count | 2035.000000 | 2035.000000 | 2035.000000 | 2035.000000 | 2035.00000 | 2.035000e+03 | 2035.000000 |
| mean | 149.713735 | 151.992826 | 147.293931 | 149.474251 | 149.45027 | 2.335681e+06 | 3899.980565 |
| std | 48.664509 | 49.413109 | 47.931958 | 48.732570 | 48.71204 | 2.091778e+06 | 4570.767877 |
| min | 81.100000 | 82.800000 | 80.000000 | 81.000000 | 80.95000 | 3.961000e+04 | 37.040000 |
| 25% | 120.025000 | 122.100000 | 118.300000 | 120.075000 | 120.05000 | 1.146444e+06 | 1427.460000 |
| 50% | 141.500000 | 143.400000 | 139.600000 | 141.100000 | 141.25000 | 1.783456e+06 | 2512.030000 |
| 75% | 157.175000 | 159.400000 | 155.150000 | 156.925000 | 156.90000 | 2.813594e+06 | 4539.015000 |
| max | 327.700000 | 328.750000 | 321.650000 | 325.950000 | 325.75000 | 2.919102e+07 | 55755.080000 |

### 4. Parametric Visualization

In [20]:
```
plt.figure(figsize=(10,6))
df['Close'].plot(kind='line',figsize=(16,7),color='b',label="Closing Price")

plt.ylabel("Price")
plt.legend(loc="upper right")
plt.title("Change in closing price over the years")
plt.grid()
```



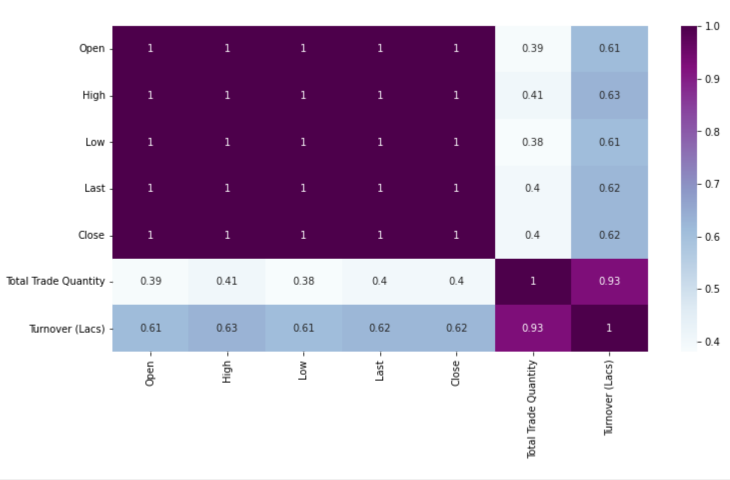

In [21]:
```
plt.figure(figsize=(10,6))
df['Open'].plot(kind='line',figsize=(16,7),color='g',label="Opening Price")

plt.ylabel("Price")
plt.legend(loc="upper left")
plt.title("Change in opening price over the years")
plt.grid()
```


In [15]:
```
df1=df.reset_index()['Close']
df1
```

Out[15]:
```
0       233.75
1       233.25
2       234.25
3       236.10
4       233.30
         ...
2030    118.65
2031    117.60
2032    120.65
2033    120.90
2034    121.55
Name: Close, Length: 2035, dtype: float64
```

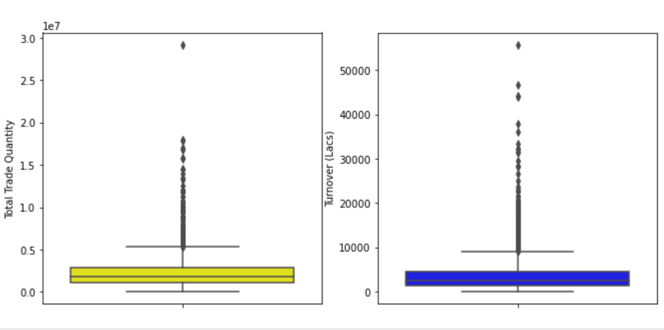
In [22]:
```
plt.figure(figsize=(12,6))
sns.heatmap(df.corr(),annot=True,cmap='BuPu')
```

Out[22]:
```
<AxesSubplot:>
```




In [19]:
```
plt.figure(figsize=(11,5))
plt.subplot(1,2,1)
sns.boxplot(data=df,y='Total Trade Quantity',color='yellow')
plt.subplot(1,2,2)
sns.boxplot(data=df,y='Turnover (Lacs)',color='blue')
```

Out[19]:
```
<AxesSubplot:ylabel='Turnover (Lacs)'>
```




In [23]:
```
training_set= df[['Open']]
training_set=pd.DataFrame(training_set)
training_set
```

Out[23]:

| | Open |
|---|---|
| 0 | 234.05 |
| 1 | 234.55 |
| 2 | 240.00 |
| 3 | 233.30 |
| 4 | 233.55 |
| ... | ... |
| 2030 | 117.60 |
| 2031 | 120.10 |
| 2032 | 121.80 |
| 2033 | 120.30 |
| 2034 | 122.10 |

2035 rows × 1 columns

### 5.Splitting and Transforming the Dataset

In [24]:
```
scaler=MinMaxScaler(feature_range=(0,1))
training_set_scaler=scaler.fit_transform(np.array(df1).reshape(-1,1))
```

In [25]:
```
train_size1= int(len(training_set_scaler)*0.65)
test_size1=int(len(training_set_scaler))-train_size1
train_data1,test_data1=training_set_scaler[0:train_size1,:],training_set_scaler[train_size1:len(df),:1]
```

In [26]:
```
def create_dataset(dataset,time_step=1):
    dataX, dataY = [], []
    for i in range(len(dataset)-time_step-1):
        a = dataset[i:(i+time_step), 0]
        dataX.append(a)
        dataY.append(dataset[i + time_step, 0])
    return np.array(dataX), np.array(dataY)
```

In [27]:
```
time_step=100
x_train, y_train=create_dataset(train_data1, time_step)
x_test, y_test= create_dataset(test_data1, time_step)
```

In [28]:
```
print(x_train.shape,y_train.shape)
```
```
(1221, 100) (1221,)
```

In [29]:
```
x_train = x_train.reshape(x_train.shape[0],x_train.shape[1] , 1)
x_test = x_test.reshape(x_test.shape[0],x_test.shape[1] , 1)
```

### 6.Building the Model

In [30]:
```
model = Sequential()
model.add(LSTM(50, return_sequences=True, input_shape=(100,1)))
model.add(LSTM(50, return_sequences=True, input_shape=(100,1)))
model.add(LSTM(50))
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='adam', metrics='acc')
```

In [31]:
```
model.summary()
```
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 100, 50)           10400

 lstm_1 (LSTM)               (None, 100, 50)           20200

 lstm_2 (LSTM)               (None, 50)                20200

 dense (Dense)               (None, 1)                 51

=================================================================
Total params: 50,851
Trainable params: 50,851
Non-trainable params: 0
_________________________________________________________________
```

In [35]:
```
model.fit(x_train, y_train, validation_data = (x_test, y_test), epochs = 75, batch_size = 64, verbose = 1)
```
```
Epoch 1/75
20/20 [==============================] - 3s 155ms/step - loss: 2.0331e-04 - acc: 8.1900e-04 - val_loss: 1.5040e-04 - val_acc: 0.0016
Epoch 2/75
20/20 [==============================] - 3s 162ms/step - loss: 2.3441e-04 - acc: 8.1900e-04 - val_loss: 1.6225e-04 - val_acc: 0.0016
Epoch 3/75
20/20 [==============================] - 3s 163ms/step - loss: 2.9667e-04 - acc: 8.1900e-04 - val_loss: 3.2529e-04 - val_acc: 0.0016
Epoch 4/75
20/20 [==============================] - 3s 156ms/step - loss: 2.4582e-04 - acc: 8.1900e-04 - val_loss: 1.5561e-04 - val_acc: 0.0016
Epoch 5/75
20/20 [==============================] - 3s 153ms/step - loss: 2.8293e-04 - acc: 8.1900e-04 - val_loss: 2.7323e-04 - val_acc: 0.0016
Epoch 6/75
20/20 [==============================] - 3s 144ms/step - loss: 2.0073e-04 - acc: 8.1900e-04 - val_loss: 1.9894e-04 - val_acc: 0.0016
Epoch 7/75
20/20 [==============================] - 3s 144ms/step - loss: 1.9292e-04 - acc: 8.1900e-04 - val_loss: 1.8148e-04 - val_acc: 0.0016
Epoch 8/75
20/20 [==============================] - 3s 144ms/step - loss: 2.0826e-04 - acc: 8.1900e-04 - val_loss: 1.5268e-04 - val_acc: 0.0016
Epoch 9/75
20/20 [==============================] - 3s 154ms/step - loss: 2.0269e-04 - acc: 8.1900e-04 - val_loss: 1.8278e-04 - val_acc: 0.0016
Epoch 10/75
20/20 [==============================] - 3s 167ms/step - loss: 2.0109e-04 - acc: 8.1900e-04 - val_loss: 1.3896e-04 - val_acc: 0.0016
Epoch 11/75
20/20 [==============================] - 4s 196ms/step - loss: 2.0636e-04 - acc: 8.1900e-04 - val_loss: 1.6420e-04 - val_acc: 0.0016
Epoch 12/75
20/20 [==============================] - 4s 184ms/step - loss: 1.9630e-04 - acc: 8.1900e-04 - val_loss: 1.5510e-04 - val_acc: 0.0016
Epoch 13/75
20/20 [==============================] - 4s 186ms/step - loss: 1.9333e-04 - acc: 8.1900e-04 - val_loss: 1.9411e-04 - val_acc: 0.0016
Epoch 14/75
20/20 [==============================] - 3s 169ms/step - loss: 2.4259e-04 - acc: 8.1900e-04 - val_loss: 1.5472e-04 - val_acc: 0.0016
Epoch 15/75
20/20 [==============================] - 4s 181ms/step - loss: 1.9696e-04 - acc: 8.1900e-04 - val_loss: 2.3213e-04 - val_acc: 0.0016
Epoch 16/75
20/20 [==============================] - 4s 180ms/step - loss: 2.0403e-04 - acc: 8.1900e-04 - val_loss: 1.6398e-04 - val_acc: 0.0016
Epoch 17/75
20/20 [==============================] - 4s 177ms/step - loss: 2.1027e-04 - acc: 8.1900e-04 - val_loss: 1.3991e-04 - val_acc: 0.0016
Epoch 18/75
20/20 [==============================] - 4s 180ms/step - loss: 1.8791e-04 - acc: 8.1900e-04 - val_loss: 1.9755e-04 - val_acc: 0.0016
Epoch 19/75
20/20 [==============================] - 4s 183ms/step - loss: 2.1035e-04 - acc: 8.1900e-04 - val_loss: 1.5320e-04 - val_acc: 0.0016
Epoch 20/75
20/20 [==============================] - 4s 180ms/step - loss: 1.9628e-04 - acc: 8.1900e-04 - val_loss: 1.7183e-04 - val_acc: 0.0016
Epoch 21/75
20/20 [==============================] - 4s 181ms/step - loss: 2.7657e-04 - acc: 8.1900e-04 - val_loss: 2.0079e-04 - val_acc: 0.0016
Epoch 22/75
20/20 [==============================] - 4s 192ms/step - loss: 2.5389e-04 - acc: 8.1900e-04 - val_loss: 1.5079e-04 - val_acc: 0.0016
Epoch 23/75
20/20 [==============================] - 4s 180ms/step - loss: 1.7738e-04 - acc: 8.1900e-04 - val_loss: 1.9378e-04 - val_acc: 0.0016
Epoch 24/75
20/20 [==============================] - 4s 179ms/step - loss: 1.8786e-04 - acc: 8.1900e-04 - val_loss: 1.5030e-04 - val_acc: 0.0016
Epoch 25/75
20/20 [==============================] - 4s 181ms/step - loss: 1.8655e-04 - acc: 8.1900e-04 - val_loss: 2.2632e-04 - val_acc: 0.0016
Epoch 26/75
20/20 [==============================] - 4s 195ms/step - loss: 1.9684e-04 - acc: 8.1900e-04 - val_loss: 2.5431e-04 - val_acc: 0.0016
Epoch 27/75
20/20 [==============================] - 4s 196ms/step - loss: 2.3503e-04 - acc: 8.1900e-04 - val_loss: 1.5269e-04 - val_acc: 0.0016
Epoch 28/75
20/20 [==============================] - 3s 171ms/step - loss: 2.0298e-04 - acc: 8.1900e-04 - val_loss: 1.7677e-04 - val_acc: 0.0016
Epoch 29/75
20/20 [==============================] - 4s 178ms/step - loss: 1.9784e-04 - acc: 8.1900e-04 - val_loss: 2.2517e-04 - val_acc: 0.0016
Epoch 30/75
20/20 [==============================] - 4s 180ms/step - loss: 2.4175e-04 - acc: 8.1900e-04 - val_loss: 1.3180e-04 - val_acc: 0.0016
Epoch 31/75
20/20 [==============================] - 4s 179ms/step - loss: 2.0865e-04 - acc: 8.1900e-04 - val_loss: 1.8257e-04 - val_acc: 0.0016
Epoch 32/75
20/20 [==============================] - 3s 171ms/step - loss: 1.7982e-04 - acc: 8.1900e-04 - val_loss: 2.2106e-04 - val_acc: 0.0016
Epoch 33/75
20/20 [==============================] - 3s 170ms/step - loss: 1.9462e-04 - acc: 8.1900e-04 - val_loss: 1.7760e-04 - val_acc: 0.0016
Epoch 34/75
20/20 [==============================] - 3s 170ms/step - loss: 1.8402e-04 - acc: 8.1900e-04 - val_loss: 1.8723e-04 - val_acc: 0.0016
Epoch 35/75
20/20 [==============================] - 4s 182ms/step - loss: 2.0072e-04 - acc: 8.1900e-04 - val_loss: 1.7638e-04 - val_acc: 0.0016
Epoch 36/75
20/20 [==============================] - 4s 186ms/step - loss: 2.0169e-04 - acc: 8.1900e-04 - val_loss: 1.7383e-04 - val_acc: 0.0016
Epoch 37/75
20/20 [==============================] - 4s 180ms/step - loss: 1.9618e-04 - acc: 8.1900e-04 - val_loss: 1.5549e-04 - val_acc: 0.0016
Epoch 38/75
20/20 [==============================] - 4s 179ms/step - loss: 1.8140e-04 - acc: 8.1900e-04 - val_loss: 1.4878e-04 - val_acc: 0.0016
Epoch 39/75
20/20 [==============================] - 4s 182ms/step - loss: 1.8933e-04 - acc: 8.1900e-04 - val_loss: 2.3624e-04 - val_acc: 0.0016
Epoch 40/75
20/20 [==============================] - 4s 197ms/step - loss: 2.1465e-04 - acc: 8.1900e-04 - val_loss: 1.5814e-04 - val_acc: 0.0016
Epoch 41/75
20/20 [==============================] - 4s 182ms/step - loss: 1.7526e-04 - acc: 8.1900e-04 - val_loss: 1.7339e-04 - val_acc: 0.0016
Epoch 42/75
20/20 [==============================] - 4s 225ms/step - loss: 1.8068e-04 - acc: 8.1900e-04 - val_loss: 2.0965e-04 - val_acc: 0.0016
Epoch 43/75
20/20 [==============================] - 4s 188ms/step - loss: 1.9956e-04 - acc: 8.1900e-04 - val_loss: 1.8579e-04 - val_acc: 0.0016
Epoch 44/75
20/20 [==============================] - 4s 204ms/step - loss: 2.0194e-04 - acc: 8.1900e-04 - val_loss: 1.4477e-04 - val_acc: 0.0016
Epoch 45/75
20/20 [==============================] - 4s 180ms/step - loss: 2.0665e-04 - acc: 8.1900e-04 - val_loss: 1.5591e-04 - val_acc: 0.0016
Epoch 46/75
20/20 [==============================] - 4s 183ms/step - loss: 1.8659e-04 - acc: 8.1900e-04 - val_loss: 1.7698e-04 - val_acc: 0.0016
Epoch 47/75
20/20 [==============================] - 4s 180ms/step - loss: 2.0381e-04 - acc: 8.1900e-04 - val_loss: 1.5979e-04 - val_acc: 0.0016
Epoch 48/75
20/20 [==============================] - 4s 179ms/step - loss: 1.9798e-04 - acc: 8.1900e-04 - val_loss: 1.4714e-04 - val_acc: 0.0016
Epoch 49/75
20/20 [==============================] - 4s 183ms/step - loss: 1.9228e-04 - acc: 8.1900e-04 - val_loss: 1.4543e-04 - val_acc: 0.0016
Epoch 50/75
20/20 [==============================] - 4s 182ms/step - loss: 1.9196e-04 - acc: 8.1900e-04 - val_loss: 1.7021e-04 - val_acc: 0.0016
Epoch 51/75
20/20 [==============================] - 4s 196ms/step - loss: 1.7998e-04 - acc: 8.1900e-04 - val_loss: 1.8265e-04 - val_acc: 0.0016
Epoch 52/75
20/20 [==============================] - 5s 223ms/step - loss: 1.9583e-04 - acc: 8.1900e-04 - val_loss: 1.7082e-04 - val_acc: 0.0016
Epoch 53/75
20/20 [==============================] - 4s 184ms/step - loss: 1.9647e-04 - acc: 8.1900e-04 - val_loss: 1.4416e-04 - val_acc: 0.0016
Epoch 54/75
20/20 [==============================] - 4s 187ms/step - loss: 1.9476e-04 - acc: 8.1900e-04 - val_loss: 1.5607e-04 - val_acc: 0.0016
Epoch 55/75
20/20 [==============================] - 4s 188ms/step - loss: 2.1146e-04 - acc: 8.1900e-04 - val_loss: 1.4861e-04 - val_acc: 0.0016
Epoch 56/75
20/20 [==============================] - 4s 182ms/step - loss: 2.4051e-04 - acc: 8.1900e-04 - val_loss: 1.3850e-04 - val_acc: 0.0016
Epoch 57/75
20/20 [==============================] - 4s 185ms/step - loss: 1.9536e-04 - acc: 8.1900e-04 - val_loss: 1.3315e-04 - val_acc: 0.0016
Epoch 58/75
20/20 [==============================] - 4s 188ms/step - loss: 2.0377e-04 - acc: 8.1900e-04 - val_loss: 1.3724e-04 - val_acc: 0.0016
Epoch 59/75
20/20 [==============================] - 3s 173ms/step - loss: 1.8622e-04 - acc: 8.1900e-04 - val_loss: 1.4796e-04 - val_acc: 0.0016
Epoch 60/75
20/20 [==============================] - 4s 175ms/step - loss: 2.0129e-04 - acc: 8.1900e-04 - val_loss: 1.5828e-04 - val_acc: 0.0016
Epoch 61/75
20/20 [==============================] - 4s 190ms/step - loss: 1.9114e-04 - acc: 8.1900e-04 - val_loss: 1.3701e-04 - val_acc: 0.0016
Epoch 62/75
20/20 [==============================] - 4s 183ms/step - loss: 1.7804e-04 - acc: 8.1900e-04 - val_loss: 1.8226e-04 - val_acc: 0.0016
Epoch 63/75
20/20 [==============================] - 5s 229ms/step - loss: 1.8138e-04 - acc: 8.1900e-04 - val_loss: 1.3514e-04 - val_acc: 0.0016
Epoch 64/75
20/20 [==============================] - 4s 207ms/step - loss: 1.8288e-04 - acc: 8.1900e-04 - val_loss: 1.5107e-04 - val_acc: 0.0016
Epoch 65/75
20/20 [==============================] - 3s 175ms/step - loss: 1.8357e-04 - acc: 8.1900e-04 - val_loss: 1.3050e-04 - val_acc: 0.0016
Epoch 66/75
20/20 [==============================] - 3s 171ms/step - loss: 1.9058e-04 - acc: 8.1900e-04 - val_loss: 1.4607e-04 - val_acc: 0.0016
Epoch 67/75
20/20 [==============================] - 3s 175ms/step - loss: 2.2311e-04 - acc: 8.1900e-04 - val_loss: 1.5109e-04 - val_acc: 0.0016
Epoch 68/75
20/20 [==============================] - 4s 194ms/step - loss: 2.4314e-04 - acc: 8.1900e-04 - val_loss: 1.6032e-04 - val_acc: 0.0016
Epoch 69/75
20/20 [==============================] - 4s 193ms/step - loss: 1.8473e-04 - acc: 8.1900e-04 - val_loss: 1.4518e-04 - val_acc: 0.0016
Epoch 70/75
20/20 [==============================] - 4s 184ms/step - loss: 1.8298e-04 - acc: 8.1900e-04 - val_loss: 1.3801e-04 - val_acc: 0.0016
Epoch 71/75
20/20 [==============================] - 4s 185ms/step - loss: 2.0626e-04 - acc: 8.1900e-04 - val_loss: 1.4628e-04 - val_acc: 0.0016
Epoch 72/75
20/20 [==============================] - 4s 175ms/step - loss: 1.9316e-04 - acc: 8.1900e-04 - val_loss: 2.0662e-04 - val_acc: 0.0016
Epoch 73/75
20/20 [==============================] - 3s 174ms/step - loss: 2.0921e-04 - acc: 8.1900e-04 - val_loss: 1.4830e-04 - val_acc: 0.0016
Epoch 74/75
20/20 [==============================] - 4s 184ms/step - loss: 2.0200e-04 - acc: 8.1900e-04 - val_loss: 1.7580e-04 - val_acc: 0.0016
Epoch 75/75
20/20 [==============================] - 4s 188ms/step - loss: 1.8000e-04 - acc: 8.1900e-04 - val_loss: 1.6119e-04 - val_acc: 0.0016
```

Out[35]:
```
<keras.callbacks.History at 0x21daa7aeac0>
```

### 7.Transforming Back to login

In [36]:
```
train_predict1=model.predict(x_train)
test_predict1=model.predict(x_test)
train_predict1=scaler.inverse_transform(train_predict1)
test_predict1=scaler.inverse_transform(test_predict1)
```
```
39/39 [==============================] - 2s 34ms/step
20/20 [==============================] - 1s 32ms/step
```

In [37]:
```
math.sqrt(mean_squared_error(y_train,train_predict1))
```

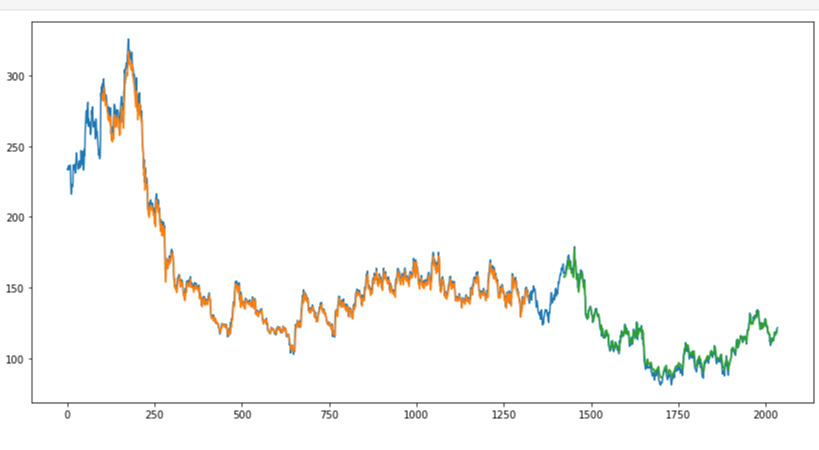
Out[37]:
```
164.07744063465765
```

In [38]:
```
math.sqrt(mean_squared_error(y_test,test_predict1))
```

Out[38]:
```
115.36804424076837
```

### 8.Plotting

In [40]:
```
# shift train predictions for plotting
look_back=100
trainPredictPlot = np.empty_like(training_set_scaler)
trainPredictPlot[:, :] = np.nan
trainPredictPlot[look_back:len(train_predict1)+look_back, :] = train_predict1

# shift test predictions for plotting
testPredictPlot = np.empty_like(training_set_scaler)
testPredictPlot[:, :] = np.nan
testPredictPlot[len(train_predict1)+(look_back*2)+1:len(df1)-1, :] = test_predict1

# plot baseline and predictions
plt.figure(figsize=(14,7))
plt.plot(scaler.inverse_transform(training_set_scaler))
plt.plot(trainPredictPlot)
plt.plot(testPredictPlot)
plt.show()
```




## THANK YOU !