Importing Libraries

In [1]:
```python
import math
import numpy as np
import pandas as pd
import seaborn as sns
sns.set_style('whitegrid')
import matplotlib.pyplot as plt
plt.style.use("fivethirtyeight")

import keras
from keras.models import Sequential
from keras.callbacks import EarlyStopping
from keras.layers import Dense, LSTM, Dropout

from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, mean_absolute_error
```

Importing DataSet

In [2]:
```python
data_dir = 'BTC-INR.csv'
df = pd.read_csv(data_dir)
```

Exploring DataSet

In [3]:
```python
df.head()
```

Out[3]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	2022-11-27	1344603.750	1355248.750	1342395.625	1343016.500	1343016.500	1669633033541
1	2022-11-28	1342656.750	1346144.875	1311117.125	1324153.250	1324153.250	2265233262656
2	2022-11-29	1324179.250	1349349.750	1317790.625	1342422.125	1342422.125	1924998924567
3	2022-11-30	1342462.375	1397715.125	1342462.375	1395897.875	1395897.875	2400427532080
4	2022-12-01	1395852.000	1395602.500	1370534.875	1377094.000	1377094.000	1858246065556

In [4]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 366 entries, 0 to 365
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Date       366 non-null    object
 1   Open       366 non-null    float64
 2   High       366 non-null    float64
 3   Low        366 non-null    float64
 4   Close      366 non-null    float64
 5   Adj Close  366 non-null    float64
 6   Volume     366 non-null    int64
dtypes: float64(5), int64(1), object(1)
memory usage: 20.1+ KB
```

DATA PREPROCESSING

Create a dataframe with date and close columns

In [5]:
```python
df = df.groupby('Date')['Close'].mean()
df = pd.DataFrame(df)
df.head()
```

Out[5]:

Date	Close
2022-11-27	1343016.500
2022-11-28	1324153.250
2022-11-29	1342422.125
2022-11-30	1395897.875
2022-12-01	1377094.000

In [6]:
```python
df.shape
```

Out[6]:
```
(366, 1)
```

NORMALIZING DATA

In [7]:
```python
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler(feature_range= (0, 1))
scaled_data = scaler.fit_transform(np.array(df.values).reshape(-1, 1))
```

SPLITTING THE DATA

In [8]:
```python
# 75% to Train , 25% to Test
train_size = int(len(df)*.75)
test_size = len(df) - train_size

print("Train Size :",train_size,"Test Size :",test_size)

train_data = scaled_data[ :train_size , 0:1 ]
test_data = scaled_data[ train_size-60: , 0:1 ]
```

```
Train Size : 274 Test Size : 92
```

In [9]:
```python
train_data.shape, test_data.shape
```

Out[9]:
```
((274, 1), (152, 1))
```

Creating Training DataSet

In [10]:
```python
# Creating a Training set with 60 time-steps and 1 output
x_train = []
y_train = []

for i in range(60, len(train_data)):
    x_train.append(train_data[i-60:i, 0])
    y_train.append(train_data[i, 0])
```

In [11]:
```python
# Convert to numpy array
x_train, y_train = np.array(x_train), np.array(y_train)
```

In [12]:
```python
# Reshaping the input
x_train = np.reshape(x_train, (x_train.shape[0], x_train.shape[1], 1))
```

In [13]:
```python
x_train.shape , y_train.shape
```

Out[13]:
```
((214, 60, 1), (214,))
```

LSTM MODEL

MODEL STRUCTURE

In [14]:
```python
model = Sequential([
    LSTM(50, return_sequences= True, input_shape= (x_train.shape[1], 1)),
    LSTM(64, return_sequences= False),
    Dense(32),
    Dense(16),
    Dense(1)
])

model.compile(optimizer= 'adam', loss= 'mse' , metrics= "mean_absolute_error")
```

In [15]:
```python
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 60, 50)            10400

 lstm_1 (LSTM)               (None, 64)                29440

 dense (Dense)               (None, 32)                2080

 dense_1 (Dense)             (None, 16)                528

 dense_2 (Dense)             (None, 1)                 17

=================================================================
Total params: 42,465
Trainable params: 42,465
Non-trainable params: 0
_________________________________________________________________
```

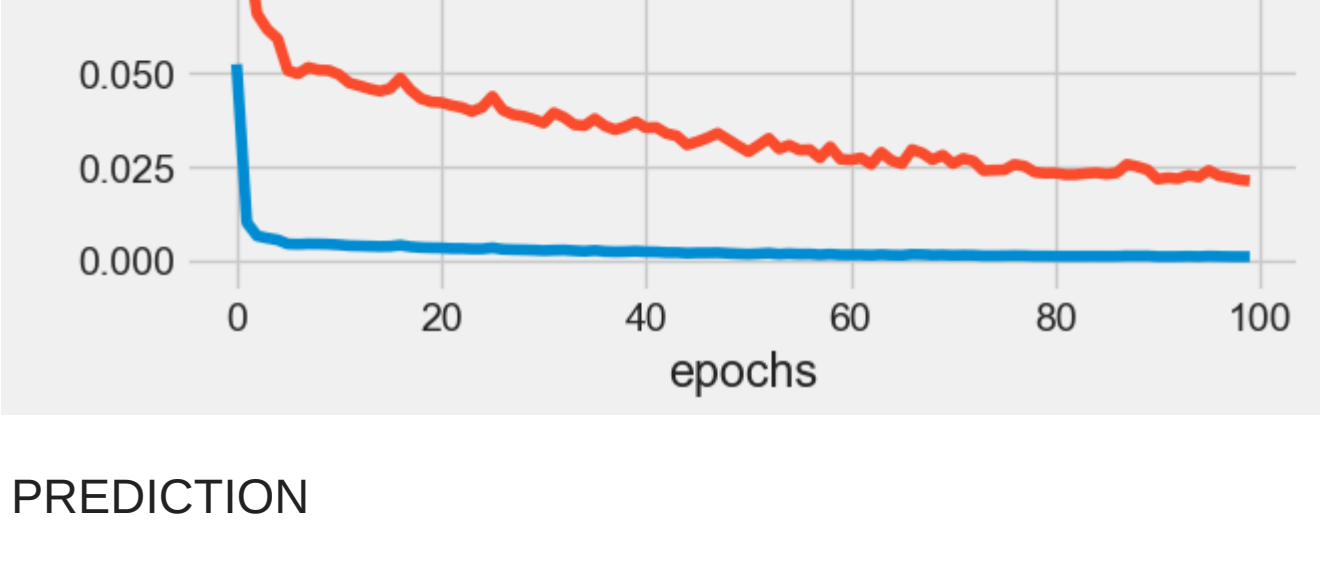
MODEL TRAINING

In [16]:
```python
# Fitting the LSTM to the Training set
callbacks = [EarlyStopping(monitor= 'loss', patience= 10 , restore_best_weights= True
history = model.fit(x_train, y_train, epochs= 100, batch_size= 32 , callbacks= callba
```

```
Epoch 1/100
2023-11-28 00:14:43.116427: W tensorflow/tsl/platform/profile_utils/cpu_utils.cc:128] Failed to get CPU frequency: 0 Hz
7/7 [==============================] - 2s 35ms/step - loss: 0.0525 - mean_absolute_error: 0.1905
Epoch 2/100
7/7 [==============================] - 0s 32ms/step - loss: 0.0101 - mean_absolute_error: 0.0843
Epoch 3/100
7/7 [==============================] - 0s 32ms/step - loss: 0.0065 - mean_absolute_error: 0.0660
Epoch 4/100
7/7 [==============================] - 0s 33ms/step - loss: 0.0059 - mean_absolute_error: 0.0618
Epoch 5/100
7/7 [==============================] - 0s 31ms/step - loss: 0.0054 - mean_absolute_error: 0.0593
Epoch 6/100
7/7 [==============================] - 0s 32ms/step - loss: 0.0043 - mean_absolute_error: 0.0508
Epoch 7/100
7/7 [==============================] - 0s 32ms/step - loss: 0.0042 - mean_absolute_error: 0.0499
Epoch 8/100
7/7 [==============================] - 0s 31ms/step - loss: 0.0044 - mean_absolute_error: 0.0516
Epoch 9/100
7/7 [==============================] - 0s 32ms/step - loss: 0.0043 - mean_absolute_error: 0.0509
Epoch 10/100
7/7 [==============================] - 0s 32ms/step - loss: 0.0043 - mean_absolute_error: 0.0508
Epoch 11/100
7/7 [==============================] - 0s 32ms/step - loss: 0.0041 - mean_absolute_error: 0.0497
Epoch 12/100
7/7 [==============================] - 0s 32ms/step - loss: 0.0039 - mean_absolute_error: 0.0475
Epoch 13/100
7/7 [==============================] - 0s 32ms/step - loss: 0.0038 - mean_absolute_error: 0.0467
Epoch 14/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0037 - mean_absolute_error: 0.0459
Epoch 15/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0036 - mean_absolute_error: 0.0453
Epoch 16/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0037 - mean_absolute_error: 0.0460
Epoch 17/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0040 - mean_absolute_error: 0.0488
Epoch 18/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0036 - mean_absolute_error: 0.0455
Epoch 19/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0034 - mean_absolute_error: 0.0433
Epoch 20/100
7/7 [==============================] - 0s 36ms/step - loss: 0.0033 - mean_absolute_error: 0.0424
Epoch 21/100
7/7 [==============================] - 0s 36ms/step - loss: 0.0033 - mean_absolute_error: 0.0422
Epoch 22/100
7/7 [==============================] - 0s 36ms/step - loss: 0.0031 - mean_absolute_error: 0.0414
Epoch 23/100
7/7 [==============================] - 0s 38ms/step - loss: 0.0031 - mean_absolute_error: 0.0409
Epoch 24/100
7/7 [==============================] - 0s 36ms/step - loss: 0.0030 - mean_absolute_error: 0.0398
Epoch 25/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0030 - mean_absolute_error: 0.0409
Epoch 26/100
7/7 [==============================] - 0s 36ms/step - loss: 0.0033 - mean_absolute_error: 0.0439
Epoch 27/100
7/7 [==============================] - 0s 36ms/step - loss: 0.0029 - mean_absolute_error: 0.0402
Epoch 28/100
7/7 [==============================] - 0s 37ms/step - loss: 0.0028 - mean_absolute_error: 0.0390
Epoch 29/100
7/7 [==============================] - 0s 36ms/step - loss: 0.0027 - mean_absolute_error: 0.0385
Epoch 30/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0027 - mean_absolute_error: 0.0377
Epoch 31/100
7/7 [==============================] - 0s 36ms/step - loss: 0.0025 - mean_absolute_error: 0.0367
Epoch 32/100
7/7 [==============================] - 0s 38ms/step - loss: 0.0026 - mean_absolute_error: 0.0395
Epoch 33/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0027 - mean_absolute_error: 0.0382
Epoch 34/100
7/7 [==============================] - 0s 36ms/step - loss: 0.0025 - mean_absolute_error: 0.0363
Epoch 35/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0024 - mean_absolute_error: 0.0360
Epoch 36/100
7/7 [==============================] - 0s 46ms/step - loss: 0.0026 - mean_absolute_error: 0.0379
Epoch 37/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0023 - mean_absolute_error: 0.0359
Epoch 38/100
7/7 [==============================] - 0s 36ms/step - loss: 0.0022 - mean_absolute_error: 0.0349
Epoch 39/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0023 - mean_absolute_error: 0.0358
Epoch 40/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0024 - mean_absolute_error: 0.0370
Epoch 41/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0022 - mean_absolute_error: 0.0354
Epoch 42/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0022 - mean_absolute_error: 0.0356
Epoch 43/100
7/7 [==============================] - 0s 36ms/step - loss: 0.0020 - mean_absolute_error: 0.0339
Epoch 44/100
7/7 [==============================] - 0s 34ms/step - loss: 0.0020 - mean_absolute_error: 0.0333
Epoch 45/100
7/7 [==============================] - 0s 36ms/step - loss: 0.0018 - mean_absolute_error: 0.0309
Epoch 46/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0019 - mean_absolute_error: 0.0317
Epoch 47/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0019 - mean_absolute_error: 0.0327
Epoch 48/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0020 - mean_absolute_error: 0.0340
Epoch 49/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0018 - mean_absolute_error: 0.0323
Epoch 50/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0017 - mean_absolute_error: 0.0306
Epoch 51/100
7/7 [==============================] - 0s 36ms/step - loss: 0.0016 - mean_absolute_error: 0.0290
Epoch 52/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0017 - mean_absolute_error: 0.0307
Epoch 53/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0019 - mean_absolute_error: 0.0326
Epoch 54/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0016 - mean_absolute_error: 0.0298
Epoch 55/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0017 - mean_absolute_error: 0.0308
Epoch 56/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0016 - mean_absolute_error: 0.0295
Epoch 57/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0016 - mean_absolute_error: 0.0296
Epoch 58/100
7/7 [==============================] - 0s 36ms/step - loss: 0.0014 - mean_absolute_error: 0.0274
Epoch 59/100
7/7 [==============================] - 0s 36ms/step - loss: 0.0016 - mean_absolute_error: 0.0304
Epoch 60/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0014 - mean_absolute_error: 0.0271
Epoch 61/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0014 - mean_absolute_error: 0.0268
Epoch 62/100
7/7 [==============================] - 0s 36ms/step - loss: 0.0014 - mean_absolute_error: 0.0274
Epoch 63/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0013 - mean_absolute_error: 0.0257
Epoch 64/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0015 - mean_absolute_error: 0.0289
Epoch 65/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0013 - mean_absolute_error: 0.0267
Epoch 66/100
7/7 [==============================] - 0s 36ms/step - loss: 0.0013 - mean_absolute_error: 0.0258
Epoch 67/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0015 - mean_absolute_error: 0.0296
Epoch 68/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0015 - mean_absolute_error: 0.0287
Epoch 69/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0013 - mean_absolute_error: 0.0269
Epoch 70/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0014 - mean_absolute_error: 0.0281
Epoch 71/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0013 - mean_absolute_error: 0.0258
Epoch 72/100
7/7 [==============================] - 0s 36ms/step - loss: 0.0013 - mean_absolute_error: 0.0272
Epoch 73/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0013 - mean_absolute_error: 0.0265
Epoch 74/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0011 - mean_absolute_error: 0.0239
Epoch 75/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0011 - mean_absolute_error: 0.0241
Epoch 76/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0012 - mean_absolute_error: 0.0241
Epoch 77/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0012 - mean_absolute_error: 0.0256
Epoch 78/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0012 - mean_absolute_error: 0.0252
Epoch 79/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0011 - mean_absolute_error: 0.0235
Epoch 80/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0011 - mean_absolute_error: 0.0233
Epoch 81/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0010 - mean_absolute_error: 0.0233
Epoch 82/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0010 - mean_absolute_error: 0.0229
Epoch 83/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0010 - mean_absolute_error: 0.0229
Epoch 84/100
7/7 [==============================] - 0s 36ms/step - loss: 0.0010 - mean_absolute_error: 0.0232
Epoch 85/100
7/7 [==============================] - 0s 37ms/step - loss: 0.0010 - mean_absolute_error: 0.0234
Epoch 86/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0010 - mean_absolute_error: 0.0230
Epoch 87/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0010 - mean_absolute_error: 0.0234
Epoch 88/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0011 - mean_absolute_error: 0.0257
Epoch 89/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0011 - mean_absolute_error: 0.0250
Epoch 90/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0011 - mean_absolute_error: 0.0242
Epoch 91/100
7/7 [==============================] - 0s 35ms/step - loss: 9.5222e-04 - mean_absolute_error: 0.0217
Epoch 92/100
7/7 [==============================] - 0s 35ms/step - loss: 9.5247e-04 - mean_absolute_error: 0.0220
Epoch 93/100
7/7 [==============================] - 0s 35ms/step - loss: 9.5638e-04 - mean_absolute_error: 0.0218
Epoch 94/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0010 - mean_absolute_error: 0.0227
Epoch 95/100
7/7 [==============================] - 0s 35ms/step - loss: 9.4206e-04 - mean_absolute_error: 0.0222
Epoch 96/100
7/7 [==============================] - 0s 35ms/step - loss: 0.0011 - mean_absolute_error: 0.0240
Epoch 97/100
7/7 [==============================] - 0s 37ms/step - loss: 9.8850e-04 - mean_absolute_error: 0.0226
Epoch 98/100
7/7 [==============================] - 0s 36ms/step - loss: 9.3878e-04 - mean_absolute_error: 0.0221
Epoch 99/100
7/7 [==============================] - 0s 35ms/step - loss: 9.2416e-04 - mean_absolute_error: 0.0215
Epoch 100/100
7/7 [==============================] - 0s 35ms/step - loss: 9.2838e-04 - mean_absolute_error: 0.0212
```

MODEL EVALUATION

In [17]:
```python
plt.plot(history.history["loss"])
plt.plot(history.history["mean_absolute_error"])
plt.legend(['Mean Squared Error','Mean Absolute Error'])
plt.title("Losses")
plt.xlabel("epochs")
plt.ylabel("loss")
plt.show()
```



PREDICTION

TESTING SET

In [18]:
```python
# Creating a testing set with 60 time-steps and 1 output
x_test = []
y_test = []

for i in range(60, len(test_data)):
    x_test.append(test_data[i-60:i, 0])
    y_test.append(test_data[i, 0])
x_test, y_test = np.array(x_test), np.array(y_test)
x_test = np.reshape(x_test, (x_test.shape[0], x_test.shape[1], 1))
```

In [19]:
```python
x_test.shape , y_test.shape
```

Out[19]:
```
((92, 60, 1), (92,))
```

In [20]:
```python
#inverse y_test scaling
predictions = model.predict(x_test)

#inverse predictions scaling
predictions = scaler.inverse_transform(predictions)
predictions.shape
```

```
3/3 [==============================] - 0s 8ms/step
```

Out[20]:
```
(92, 1)
```

ROOT MEAN SQUARE ERROR

In [21]:
```python
#inverse y_test scaling
y_test = scaler.inverse_transform([y_test])

RMSE = np.sqrt(np.mean( y_test - predictions )**2).round(2)
RMSE
```

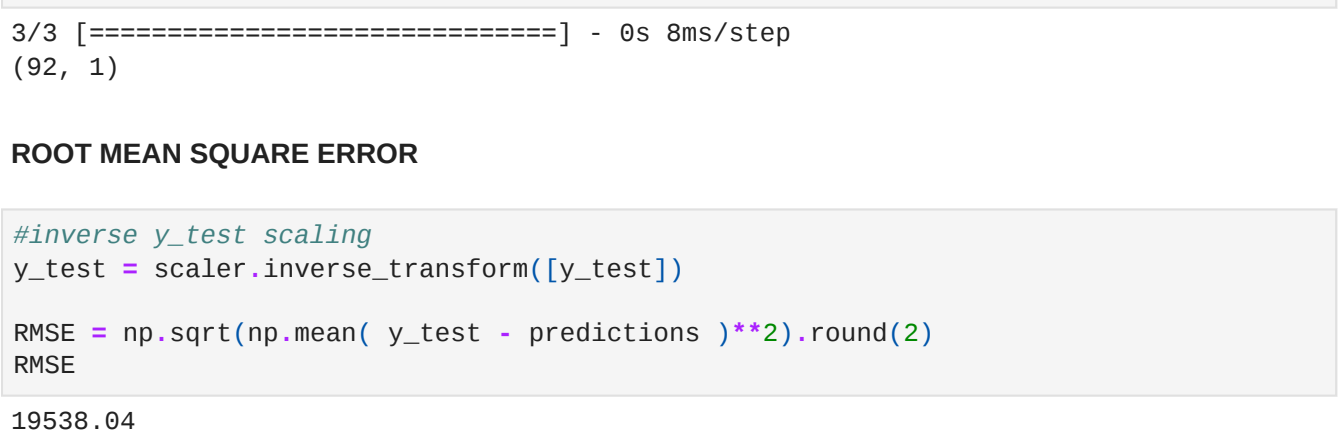
Out[21]:
```
19538.04
```

VISUALIZE PREDICTIONS WITH DATA

In [22]:
```python
train = df.iloc[:train_size , 0:1]
test = df.iloc[train_size: , 0:1]
test['Predictions'] = predictions

plt.figure(figsize= (16, 6))
plt.title('BITCOIN Stock Price Prediction', fontsize= 20)
plt.xlabel('Date', fontsize= 18)
plt.ylabel('Close Price', fontsize= 18)
plt.plot(train['Close'], linewidth= 3)
plt.plot(test['Close'], linewidth= 3)
plt.plot(test["Predictions"], linewidth= 3)
plt.legend(['Train', 'Test', 'Predictions'])
```

Out[22]:
```
<matplotlib.legend.Legend at 0x107e925c0>
```



In [23]:
```python
from datetime import timedelta
```

In [24]:
```python
def insert_end(Xin, new_input):
    timestep = 60
    for i in range(timestep - 1):
        Xin[:, i, :] = Xin[:, i+1, :]
    Xin[:, timestep - 1, :] = new_input
    return Xin
```

In [25]:
```python
future = 30
forcast = []
Xin = x_test[-1 :, :, :]
time = []
for i in range(0, future):
    out = model.predict(Xin, batch_size=5)
    forcast.append(out[0, 0])
    print(forcast)
    Xin = insert_end(Xin, out[0, 0])
    time.append(pd.to_datetime(df.index[-1]) + timedelta(days=i))
```

```
1/1 [==============================] - 0s 16ms/step
[0.9894171]
1/1 [==============================] - 0s 16ms/step
[0.9894171, 0.9835419]
1/1 [==============================] - 0s 16ms/step
[0.9894171, 0.9835419, 0.97228533]
1/1 [==============================] - 0s 15ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645]
1/1 [==============================] - 0s 15ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626]
1/1 [==============================] - 0s 16ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355]
1/1 [==============================] - 0s 17ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355, 0.9099731]
1/1 [==============================] - 0s 17ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355, 0.9099731, 0.89529437]
1/1 [==============================] - 0s 16ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355, 0.9099731, 0.89529437, 0.88228136]
1/1 [==============================] - 0s 16ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355, 0.9099731, 0.89529437, 0.88228136, 0.87114185]
1/1 [==============================] - 0s 15ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355, 0.9099731, 0.89529437, 0.88228136, 0.87114185, 0.8618616]
1/1 [==============================] - 0s 16ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355, 0.9099731, 0.89529437, 0.88228136, 0.87114185, 0.8618616, 0.85425425]
1/1 [==============================] - 0s 15ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355, 0.9099731, 0.89529437, 0.88228136, 0.87114185, 0.8618616, 0.85425425, 0.8480164]
1/1 [==============================] - 0s 16ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355, 0.9099731, 0.89529437, 0.88228136, 0.87114185, 0.8618616, 0.85425425, 0.8480164, 0.8427813]
1/1 [==============================] - 0s 15ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355, 0.9099731, 0.89529437, 0.88228136, 0.87114185, 0.8618616, 0.85425425, 0.8480164, 0.8427813, 0.83816653]
1/1 [==============================] - 0s 14ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355, 0.9099731, 0.89529437, 0.88228136, 0.87114185, 0.8618616, 0.85425425, 0.8480164, 0.8427813, 0.83816653, 0.83381385]
1/1 [==============================] - 0s 16ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355, 0.9099731, 0.89529437, 0.88228136, 0.87114185, 0.8618616, 0.85425425, 0.8480164, 0.8427813, 0.83816653, 0.83381385, 0.8294174]
1/1 [==============================] - 0s 16ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355, 0.9099731, 0.89529437, 0.88228136, 0.87114185, 0.8618616, 0.85425425, 0.8480164, 0.8427813, 0.83816653, 0.83381385, 0.8294174, 0.8247432]
1/1 [==============================] - 0s 16ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355, 0.9099731, 0.89529437, 0.88228136, 0.87114185, 0.8618616, 0.85425425, 0.8480164, 0.8427813, 0.83816653, 0.83381385, 0.8294174, 0.8247432, 0.81963843]
1/1 [==============================] - 0s 16ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355, 0.9099731, 0.89529437, 0.88228136, 0.87114185, 0.8618616, 0.85425425, 0.8480164, 0.8427813, 0.83816653, 0.83381385, 0.8294174, 0.8247432, 0.81963843, 0.8140312]
1/1 [==============================] - 0s 16ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355, 0.9099731, 0.89529437, 0.88228136, 0.87114185, 0.8618616, 0.85425425, 0.8480164, 0.8427813, 0.83816653, 0.83381385, 0.8294174, 0.8247432, 0.81963843, 0.8140312, 0.8079227]
1/1 [==============================] - 0s 15ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355, 0.9099731, 0.89529437, 0.88228136, 0.87114185, 0.8618616, 0.85425425, 0.8480164, 0.8427813, 0.83816653, 0.83381385, 0.8294174, 0.8247432, 0.81963843, 0.8140312, 0.8079227, 0.80137485]
1/1 [==============================] - 0s 15ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355, 0.9099731, 0.89529437, 0.88228136, 0.87114185, 0.8618616, 0.85425425, 0.8480164, 0.8427813, 0.83816653, 0.83381385, 0.8294174, 0.8247432, 0.81963843, 0.8140312, 0.8079227, 0.80137485, 0.79449314]
1/1 [==============================] - 0s 15ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355, 0.9099731, 0.89529437, 0.88228136, 0.87114185, 0.8618616, 0.85425425, 0.8480164, 0.8427813, 0.83816653, 0.83381385, 0.8294174, 0.8247432, 0.81963843, 0.8140312, 0.8079227, 0.80137485, 0.79449314, 0.78740895]
1/1 [==============================] - 0s 16ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355, 0.9099731, 0.89529437, 0.88228136, 0.87114185, 0.8618616, 0.85425425, 0.8480164, 0.8427813, 0.83816653, 0.83381385, 0.8294174, 0.8247432, 0.81963843, 0.8140312, 0.8079227, 0.80137485, 0.79449314, 0.78740895, 0.78026164]
1/1 [==============================] - 0s 16ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355, 0.9099731, 0.89529437, 0.88228136, 0.87114185, 0.8618616, 0.85425425, 0.8480164, 0.8427813, 0.83816653, 0.83381385, 0.8294174, 0.8247432, 0.81963843, 0.8140312, 0.8079227, 0.80137485, 0.79449314, 0.78740895, 0.78026164, 0.77318573]
1/1 [==============================] - 0s 16ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355, 0.9099731, 0.89529437, 0.88228136, 0.87114185, 0.8618616, 0.85425425, 0.8480164, 0.8427813, 0.83816653, 0.83381385, 0.8294174, 0.8247432, 0.81963843, 0.8140312, 0.8079227, 0.80137485, 0.79449314, 0.78740895, 0.78026164, 0.77318573, 0.7662973]
1/1 [==============================] - 0s 16ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355, 0.9099731, 0.89529437, 0.88228136, 0.87114185, 0.8618616, 0.85425425, 0.8480164, 0.8427813, 0.83816653, 0.83381385, 0.8294174, 0.8247432, 0.81963843, 0.8140312, 0.8079227, 0.80137485, 0.79449314, 0.78740895, 0.78026164, 0.77318573, 0.7662973, 0.75968647]
1/1 [==============================] - 0s 15ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355, 0.9099731, 0.89529437, 0.88228136, 0.87114185, 0.8618616, 0.85425425, 0.8480164, 0.8427813, 0.83816653, 0.83381385, 0.8294174, 0.8247432, 0.81963843, 0.8140312, 0.8079227, 0.80137485, 0.79449314, 0.78740895, 0.78026164, 0.77318573, 0.7662973, 0.75968647, 0.7534152]
1/1 [==============================] - 0s 15ms/step
[0.9894171, 0.9835419, 0.97228533, 0.9580645, 0.94220626, 0.92585355, 0.9099731, 0.89529437, 0.88228136, 0.87114185, 0.8618616, 0.85425425, 0.8480164, 0.8427813, 0.83816653, 0.83381385, 0.8294174, 0.8247432, 0.81963843, 0.8140312, 0.8079227, 0.80137485, 0.79449314, 0.78740895, 0.78026164, 0.77318573, 0.7662973, 0.75968647, 0.7534152, 0.7475154]
```

In [26]:
```python
time
```

Out[26]:
```
[Timestamp('2023-11-27 00:00:00'),
 Timestamp('2023-11-28 00:00:00'),
 Timestamp('2023-11-29 00:00:00'),
 Timestamp('2023-11-30 00:00:00'),
 Timestamp('2023-12-01 00:00:00'),
 Timestamp('2023-12-02 00:00:00'),
 Timestamp('2023-12-03 00:00:00'),
 Timestamp('2023-12-04 00:00:00'),
 Timestamp('2023-12-05 00:00:00'),
 Timestamp('2023-12-06 00:00:00'),
 Timestamp('2023-12-07 00:00:00'),
 Timestamp('2023-12-08 00:00:00'),
 Timestamp('2023-12-09 00:00:00'),
 Timestamp('2023-12-10 00:00:00'),
 Timestamp('2023-12-11 00:00:00'),
 Timestamp('2023-12-12 00:00:00'),
 Timestamp('2023-12-13 00:00:00'),
 Timestamp('2023-12-14 00:00:00'),
 Timestamp('2023-12-15 00:00:00'),
 Timestamp('2023-12-16 00:00:00'),
 Timestamp('2023-12-17 00:00:00'),
 Timestamp('2023-12-18 00:00:00'),
 Timestamp('2023-12-19 00:00:00'),
 Timestamp('2023-12-20 00:00:00'),
 Timestamp('2023-12-21 00:00:00'),
 Timestamp('2023-12-22 00:00:00'),
 Timestamp('2023-12-23 00:00:00'),
 Timestamp('2023-12-24 00:00:00'),
 Timestamp('2023-12-25 00:00:00'),
 Timestamp('2023-12-26 00:00:00')]
```

CREATE FORECASTED DATAFRAME

In [27]:
```python
forcasted_output = np.asanyarray(forcast)
forcasted_output = forcasted_output.reshape(-1, 1)
forcasted_output = scaler.inverse_transform(forcasted_output)
```

In [28]:
```python
forcasted_output = pd.DataFrame(forcasted_output)
date = pd.DataFrame(time)
df_result = pd.concat([date,forcasted_output], axis=1)
df_result.columns = "Date", "Forecasted"
```

FORECASTED PRICE

In [29]:
```python
df_result
```

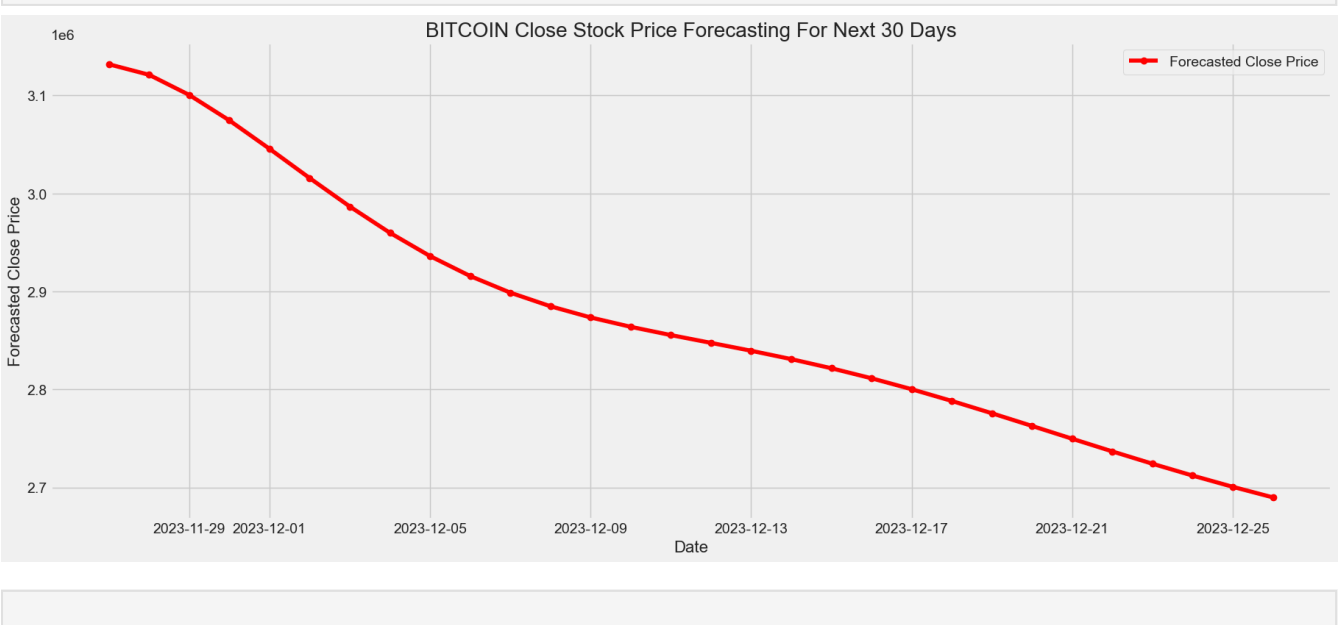
Out[29]:

	Date	Forecasted
0	2023-11-27	3131812.25
1	2023-11-28	3121078.50
2	2023-11-29	3100512.75
3	2023-11-30	3074531.50
4	2023-12-01	3045558.25
5	2023-12-02	3015682.25
6	2023-12-03	2986668.75
7	2023-12-04	2959850.75
8	2023-12-05	2936076.00
9	2023-12-06	2915724.25
10	2023-12-07	2898769.25
11	2023-12-08	2884870.50
12	2023-12-09	2873474.00
13	2023-12-10	2863909.50
14	2023-12-11	2855478.50
15	2023-12-12	2847526.25
16	2023-12-13	2839494.00
17	2023-12-14	2830954.25
18	2023-12-15	2821627.75
19	2023-12-16	2811383.50
20	2023-12-17	2800223.25
21	2023-12-18	2788260.25
22	2023-12-19	2775687.50
23	2023-12-20	2762744.50
24	2023-12-21	2749686.50
25	2023-12-22	2736758.75
26	2023-12-23	2724173.75
27	2023-12-24	2712095.75
28	2023-12-25	2700638.25
29	2023-12-26	2689859.50

In [37]:
```python
df_result['Date'] = pd.to_datetime(df_result['Date'])

plt.figure(figsize=(20, 8))
plt.title('BITCOIN Close Stock Price Forecasting For Next 30 Days')
plt.xlabel('Date', fontsize=16)
plt.ylabel('Forecasted Close Price', fontsize=16)

plt.plot(df_result['Date'], df_result['Forecasted'], marker='o',color='red', label='F
plt.legend()
plt.grid(True)
plt.show()
```



In [ ]: