In [1]:
```python
# 1.    Load the basic libraries and packages

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
from keras.models import Sequential
from keras.layers import LSTM, Dropout, Dense
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score

warnings.filterwarnings('ignore')
```

In [2]:
```python
# 2.    Load the dataset

dataset = pd.read_csv("/content/TCS Historical Data.csv")
print(dataset.head())
```

```
         Date     Price      Open      High       Low   Vol. Change %
0  28-03-2024  3,876.30  3,850.10  3,915.00  3,840.50  4.31M    0.92%
1  27-03-2024  3,840.90  3,888.50  3,895.00  3,829.40  1.97M   -0.94%
2  26-03-2024  3,877.50  3,875.00  3,946.70  3,871.45  3.44M   -0.85%
3  22-03-2024  3,910.90  3,897.00  3,938.00  3,855.00  5.85M   -1.56%
4  21-03-2024  3,972.95  3,990.05  4,008.40  3,948.00  3.83M    0.05%
```

In [3]:
```python
# 3.    Analyse the dataset

# Summary statistics
print(dataset.describe())

# Dataset info
print(dataset.info())
```

```
              Date     Price      Open      High       Low   Vol. Change %
count         1055      1055      1055      1055      1055   1055     1055
unique        1055      1045       951      1009      1014    430      469
top     28-03-2024  2,190.95  3,300.54  3,245.91  3,060.00  1.79M    0.08%
freq             1         2         3         3         3     10        8
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1055 entries, 0 to 1054
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   Date      1055 non-null   object
 1   Price     1055 non-null   object
 2   Open      1055 non-null   object
 3   High      1055 non-null   object
 4   Low       1055 non-null   object
 5   Vol.      1055 non-null   object
 6   Change %  1055 non-null   object
dtypes: object(7)
memory usage: 57.8+ KB
None
```

In [5]:
```python
# 4.    Apply LSTM Model

# Extract the feature column for modeling
tcs_training = dataset.iloc[:, 3:4].values
```

```python
# Convert the 'Price' column to numeric, removing commas
tcs_training = tcs_training.astype(str)  # Convert to string type
tcs_training = np.char.replace(tcs_training, ',', '').astype(float)  # Remove cc

# Normalize the feature using Min-Max Scaler
scaler = MinMaxScaler(feature_range=(0, 1))
tcs_training_scaled = scaler.fit_transform(tcs_training)

# Create the feature set and labels
feature_set = []
labels = []

for i in range(200, 1055):
    feature_set.append(tcs_training_scaled[i-200:i, 0])
    labels.append(tcs_training_scaled[i, 0])

# Convert to numpy arrays
feature_set = np.array(feature_set)
labels = np.array(labels)

# Reshape feature set for LSTM input
feature_set = np.reshape(feature_set, (feature_set.shape[0], feature_set.shape[1

# Initialize the model
model = Sequential()

# Add LSTM layers with Dropout
model.add(LSTM(units=60, return_sequences=True, input_shape=(feature_set.shape[1
model.add(Dropout(0.20))

model.add(LSTM(units=60, return_sequences=True))
model.add(Dropout(0.20))

model.add(LSTM(units=60, return_sequences=True))
model.add(Dropout(0.20))

model.add(LSTM(units=60))
model.add(Dropout(0.20))

# Add output layer
model.add(Dense(units=1))

# Compile the model
model.compile(optimizer='adam', loss='mean_squared_error', metrics=['accuracy'])
```

In [6]:

```python
# 5.    Apply the training over the dataset to minimize the loss

history = model.fit(feature_set, labels, epochs=100, batch_size=32, validation_s
```

```
Epoch 1/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 518ms/step - accuracy: 0.0000e+00 - loss: 0.1418 -
val_accuracy: 0.0058 - val_loss: 0.0295
Epoch 2/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 496ms/step - accuracy: 0.0000e+00 - loss: 0.0124 -
val_accuracy: 0.0058 - val_loss: 0.0226
Epoch 3/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 496ms/step - accuracy: 0.0000e+00 - loss: 0.0092 -
val_accuracy: 0.0058 - val_loss: 0.0187
Epoch 4/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 483ms/step - accuracy: 0.0000e+00 - loss: 0.0081 -
val_accuracy: 0.0058 - val_loss: 0.0146
Epoch 5/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 21s 497ms/step - accuracy: 0.0000e+00 - loss: 0.0088 -
val_accuracy: 0.0058 - val_loss: 0.0068
Epoch 6/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 496ms/step - accuracy: 0.0000e+00 - loss: 0.0066 -
val_accuracy: 0.0058 - val_loss: 0.0044
Epoch 7/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 496ms/step - accuracy: 0.0000e+00 - loss: 0.0068 -
val_accuracy: 0.0058 - val_loss: 0.0033
Epoch 8/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 460ms/step - accuracy: 0.0000e+00 - loss: 0.0059 -
val_accuracy: 0.0058 - val_loss: 0.0030
Epoch 9/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 10s 446ms/step - accuracy: 0.0000e+00 - loss: 0.0057 -
val_accuracy: 0.0058 - val_loss: 0.0043
Epoch 10/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 487ms/step - accuracy: 0.0000e+00 - loss: 0.0061 -
val_accuracy: 0.0058 - val_loss: 0.0023
Epoch 11/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 493ms/step - accuracy: 0.0000e+00 - loss: 0.0061 -
val_accuracy: 0.0058 - val_loss: 0.0042
Epoch 12/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 492ms/step - accuracy: 0.0000e+00 - loss: 0.0079 -
val_accuracy: 0.0058 - val_loss: 0.0025
Epoch 13/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 9s 429ms/step - accuracy: 0.0000e+00 - loss: 0.0060 -
val_accuracy: 0.0058 - val_loss: 0.0031
Epoch 14/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 454ms/step - accuracy: 0.0000e+00 - loss: 0.0057 -
val_accuracy: 0.0058 - val_loss: 0.0020
Epoch 15/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 491ms/step - accuracy: 0.0000e+00 - loss: 0.0052 -
val_accuracy: 0.0058 - val_loss: 0.0026
Epoch 16/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 489ms/step - accuracy: 0.0000e+00 - loss: 0.0050 -
val_accuracy: 0.0058 - val_loss: 0.0019
Epoch 17/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 19s 433ms/step - accuracy: 0.0000e+00 - loss: 0.0044 -
val_accuracy: 0.0058 - val_loss: 0.0024
Epoch 18/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 456ms/step - accuracy: 0.0000e+00 - loss: 0.0051 -
val_accuracy: 0.0058 - val_loss: 0.0022
Epoch 19/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 21s 494ms/step - accuracy: 0.0000e+00 - loss: 0.0051 -
val_accuracy: 0.0058 - val_loss: 0.0017
Epoch 20/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 493ms/step - accuracy: 0.0000e+00 - loss: 0.0045 -
val_accuracy: 0.0058 - val_loss: 0.0017
```

```
Epoch 21/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 465ms/step - accuracy: 0.0000e+00 - loss: 0.0045 -
val_accuracy: 0.0058 - val_loss: 0.0017
Epoch 22/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 496ms/step - accuracy: 0.0000e+00 - loss: 0.0041 -
val_accuracy: 0.0058 - val_loss: 0.0016
Epoch 23/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 498ms/step - accuracy: 0.0000e+00 - loss: 0.0061 -
val_accuracy: 0.0058 - val_loss: 0.0018
Epoch 24/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 472ms/step - accuracy: 0.0000e+00 - loss: 0.0040 -
val_accuracy: 0.0058 - val_loss: 0.0017
Epoch 25/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 10s 433ms/step - accuracy: 0.0000e+00 - loss: 0.0042 -
val_accuracy: 0.0058 - val_loss: 0.0018
Epoch 26/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 488ms/step - accuracy: 0.0000e+00 - loss: 0.0044 -
val_accuracy: 0.0058 - val_loss: 0.0017
Epoch 27/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 488ms/step - accuracy: 0.0000e+00 - loss: 0.0038 -
val_accuracy: 0.0058 - val_loss: 0.0020
Epoch 28/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 19s 440ms/step - accuracy: 0.0000e+00 - loss: 0.0038 -
val_accuracy: 0.0058 - val_loss: 0.0018
Epoch 29/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 444ms/step - accuracy: 0.0000e+00 - loss: 0.0039 -
val_accuracy: 0.0058 - val_loss: 0.0018
Epoch 30/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 496ms/step - accuracy: 0.0000e+00 - loss: 0.0036 -
val_accuracy: 0.0058 - val_loss: 0.0026
Epoch 31/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 492ms/step - accuracy: 0.0000e+00 - loss: 0.0038 -
val_accuracy: 0.0058 - val_loss: 0.0029
Epoch 32/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 482ms/step - accuracy: 0.0000e+00 - loss: 0.0036 -
val_accuracy: 0.0058 - val_loss: 0.0037
Epoch 33/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 9s 429ms/step - accuracy: 0.0000e+00 - loss: 0.0030 -
val_accuracy: 0.0058 - val_loss: 0.0038
Epoch 34/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 458ms/step - accuracy: 0.0000e+00 - loss: 0.0036 -
val_accuracy: 0.0058 - val_loss: 0.0027
Epoch 35/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 21s 492ms/step - accuracy: 0.0000e+00 - loss: 0.0039 -
val_accuracy: 0.0058 - val_loss: 0.0022
Epoch 36/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 485ms/step - accuracy: 0.0000e+00 - loss: 0.0034 -
val_accuracy: 0.0058 - val_loss: 0.0030
Epoch 37/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 459ms/step - accuracy: 0.0000e+00 - loss: 0.0043 -
val_accuracy: 0.0058 - val_loss: 0.0033
Epoch 38/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 495ms/step - accuracy: 0.0000e+00 - loss: 0.0037 -
val_accuracy: 0.0058 - val_loss: 0.0043
Epoch 39/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 493ms/step - accuracy: 0.0000e+00 - loss: 0.0035 -
val_accuracy: 0.0058 - val_loss: 0.0024
Epoch 40/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 19s 426ms/step - accuracy: 0.0000e+00 - loss: 0.0038 -
val_accuracy: 0.0058 - val_loss: 0.0041
```

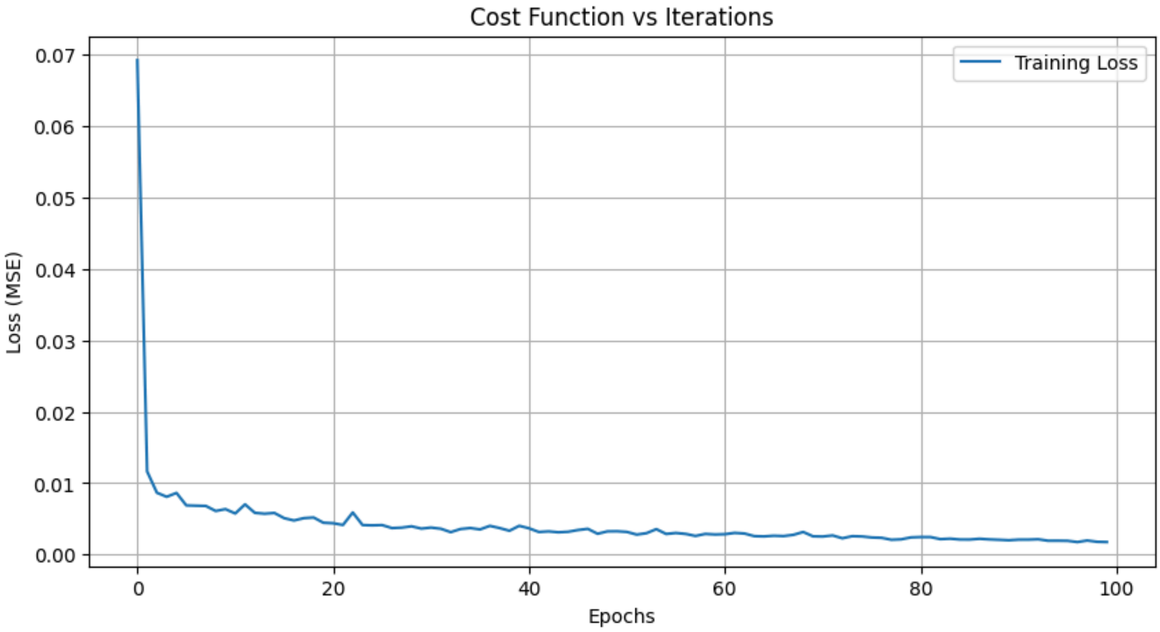
```
Epoch 41/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 492ms/step - accuracy: 0.0000e+00 - loss: 0.0035 -
val_accuracy: 0.0058 - val_loss: 0.0036
Epoch 42/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 498ms/step - accuracy: 0.0000e+00 - loss: 0.0033 -
val_accuracy: 0.0058 - val_loss: 0.0041
Epoch 43/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 21s 499ms/step - accuracy: 0.0000e+00 - loss: 0.0035 -
val_accuracy: 0.0058 - val_loss: 0.0052
Epoch 44/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 10s 471ms/step - accuracy: 0.0000e+00 - loss: 0.0033 -
val_accuracy: 0.0058 - val_loss: 0.0041
Epoch 45/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 21s 510ms/step - accuracy: 0.0000e+00 - loss: 0.0032 -
val_accuracy: 0.0058 - val_loss: 0.0048
Epoch 46/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 21s 528ms/step - accuracy: 0.0000e+00 - loss: 0.0035 -
val_accuracy: 0.0058 - val_loss: 0.0053
Epoch 47/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 19s 465ms/step - accuracy: 0.0000e+00 - loss: 0.0036 -
val_accuracy: 0.0058 - val_loss: 0.0042
Epoch 48/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 21s 509ms/step - accuracy: 0.0000e+00 - loss: 0.0034 -
val_accuracy: 0.0058 - val_loss: 0.0056
Epoch 49/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 21s 550ms/step - accuracy: 0.0000e+00 - loss: 0.0029 -
val_accuracy: 0.0058 - val_loss: 0.0046
Epoch 50/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 521ms/step - accuracy: 0.0000e+00 - loss: 0.0033 -
val_accuracy: 0.0058 - val_loss: 0.0045
Epoch 51/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 513ms/step - accuracy: 0.0000e+00 - loss: 0.0032 -
val_accuracy: 0.0058 - val_loss: 0.0062
Epoch 52/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 21s 540ms/step - accuracy: 0.0000e+00 - loss: 0.0027 -
val_accuracy: 0.0058 - val_loss: 0.0060
Epoch 53/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 12s 539ms/step - accuracy: 0.0000e+00 - loss: 0.0029 -
val_accuracy: 0.0058 - val_loss: 0.0040
Epoch 54/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 507ms/step - accuracy: 0.0000e+00 - loss: 0.0035 -
val_accuracy: 0.0058 - val_loss: 0.0085
Epoch 55/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 524ms/step - accuracy: 0.0000e+00 - loss: 0.0029 -
val_accuracy: 0.0058 - val_loss: 0.0043
Epoch 56/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 21s 562ms/step - accuracy: 0.0000e+00 - loss: 0.0028 -
val_accuracy: 0.0058 - val_loss: 0.0073
Epoch 57/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 12s 544ms/step - accuracy: 0.0000e+00 - loss: 0.0030 -
val_accuracy: 0.0058 - val_loss: 0.0067
Epoch 58/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 497ms/step - accuracy: 0.0000e+00 - loss: 0.0024 -
val_accuracy: 0.0058 - val_loss: 0.0056
Epoch 59/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 526ms/step - accuracy: 0.0000e+00 - loss: 0.0030 -
val_accuracy: 0.0058 - val_loss: 0.0076
Epoch 60/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 510ms/step - accuracy: 0.0000e+00 - loss: 0.0029 -
val_accuracy: 0.0058 - val_loss: 0.0033
```

```
Epoch 61/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 21s 519ms/step - accuracy: 0.0000e+00 - loss: 0.0027 -
val_accuracy: 0.0058 - val_loss: 0.0091
Epoch 62/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 12s 558ms/step - accuracy: 0.0000e+00 - loss: 0.0028 -
val_accuracy: 0.0058 - val_loss: 0.0065
Epoch 63/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 19s 497ms/step - accuracy: 0.0000e+00 - loss: 0.0035 -
val_accuracy: 0.0058 - val_loss: 0.0058
Epoch 64/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 21s 489ms/step - accuracy: 0.0000e+00 - loss: 0.0024 -
val_accuracy: 0.0058 - val_loss: 0.0071
Epoch 65/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 491ms/step - accuracy: 0.0000e+00 - loss: 0.0027 -
val_accuracy: 0.0058 - val_loss: 0.0074
Epoch 66/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 502ms/step - accuracy: 0.0000e+00 - loss: 0.0029 -
val_accuracy: 0.0058 - val_loss: 0.0051
Epoch 67/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 444ms/step - accuracy: 0.0000e+00 - loss: 0.0026 -
val_accuracy: 0.0058 - val_loss: 0.0055
Epoch 68/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 12s 547ms/step - accuracy: 0.0000e+00 - loss: 0.0026 -
val_accuracy: 0.0058 - val_loss: 0.0045
Epoch 69/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 506ms/step - accuracy: 0.0000e+00 - loss: 0.0031 -
val_accuracy: 0.0058 - val_loss: 0.0043
Epoch 70/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 502ms/step - accuracy: 0.0000e+00 - loss: 0.0027 -
val_accuracy: 0.0058 - val_loss: 0.0045
Epoch 71/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 21s 490ms/step - accuracy: 0.0000e+00 - loss: 0.0029 -
val_accuracy: 0.0058 - val_loss: 0.0073
Epoch 72/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 494ms/step - accuracy: 0.0000e+00 - loss: 0.0027 -
val_accuracy: 0.0058 - val_loss: 0.0041
Epoch 73/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 21s 510ms/step - accuracy: 0.0000e+00 - loss: 0.0021 -
val_accuracy: 0.0058 - val_loss: 0.0068
Epoch 74/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 469ms/step - accuracy: 0.0000e+00 - loss: 0.0026 -
val_accuracy: 0.0058 - val_loss: 0.0046
Epoch 75/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 501ms/step - accuracy: 0.0000e+00 - loss: 0.0024 -
val_accuracy: 0.0058 - val_loss: 0.0080
Epoch 76/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 12s 538ms/step - accuracy: 0.0000e+00 - loss: 0.0024 -
val_accuracy: 0.0058 - val_loss: 0.0043
Epoch 77/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 514ms/step - accuracy: 0.0000e+00 - loss: 0.0024 -
val_accuracy: 0.0058 - val_loss: 0.0054
Epoch 78/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 515ms/step - accuracy: 0.0000e+00 - loss: 0.0020 -
val_accuracy: 0.0058 - val_loss: 0.0048
Epoch 79/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 468ms/step - accuracy: 0.0000e+00 - loss: 0.0023 -
val_accuracy: 0.0058 - val_loss: 0.0044
Epoch 80/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 507ms/step - accuracy: 0.0000e+00 - loss: 0.0024 -
val_accuracy: 0.0058 - val_loss: 0.0046
```

```
Epoch 81/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 500ms/step - accuracy: 0.0000e+00 - loss: 0.0027 -
val_accuracy: 0.0058 - val_loss: 0.0038
Epoch 82/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 19s 441ms/step - accuracy: 0.0000e+00 - loss: 0.0025 -
val_accuracy: 0.0058 - val_loss: 0.0071
Epoch 83/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 490ms/step - accuracy: 0.0000e+00 - loss: 0.0022 -
val_accuracy: 0.0058 - val_loss: 0.0054
Epoch 84/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 497ms/step - accuracy: 0.0000e+00 - loss: 0.0020 -
val_accuracy: 0.0058 - val_loss: 0.0073
Epoch 85/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 468ms/step - accuracy: 0.0000e+00 - loss: 0.0022 -
val_accuracy: 0.0058 - val_loss: 0.0041
Epoch 86/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 21s 494ms/step - accuracy: 0.0000e+00 - loss: 0.0020 -
val_accuracy: 0.0058 - val_loss: 0.0079
Epoch 87/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 501ms/step - accuracy: 0.0000e+00 - loss: 0.0022 -
val_accuracy: 0.0058 - val_loss: 0.0046
Epoch 88/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 503ms/step - accuracy: 0.0000e+00 - loss: 0.0024 -
val_accuracy: 0.0058 - val_loss: 0.0067
Epoch 89/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 502ms/step - accuracy: 0.0000e+00 - loss: 0.0020 -
val_accuracy: 0.0058 - val_loss: 0.0042
Epoch 90/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 20s 462ms/step - accuracy: 0.0000e+00 - loss: 0.0022 -
val_accuracy: 0.0058 - val_loss: 0.0055
Epoch 91/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 21s 498ms/step - accuracy: 0.0000e+00 - loss: 0.0021 -
val_accuracy: 0.0058 - val_loss: 0.0061
Epoch 92/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 490ms/step - accuracy: 0.0000e+00 - loss: 0.0023 -
val_accuracy: 0.0058 - val_loss: 0.0080
Epoch 93/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 494ms/step - accuracy: 0.0000e+00 - loss: 0.0022 -
val_accuracy: 0.0058 - val_loss: 0.0052
Epoch 94/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 10s 436ms/step - accuracy: 0.0000e+00 - loss: 0.0020 -
val_accuracy: 0.0058 - val_loss: 0.0061
Epoch 95/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 460ms/step - accuracy: 0.0000e+00 - loss: 0.0019 -
val_accuracy: 0.0058 - val_loss: 0.0052
Epoch 96/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 21s 496ms/step - accuracy: 0.0000e+00 - loss: 0.0019 -
val_accuracy: 0.0058 - val_loss: 0.0032
Epoch 97/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 14s 663ms/step - accuracy: 0.0000e+00 - loss: 0.0018 -
val_accuracy: 0.0058 - val_loss: 0.0042
Epoch 98/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 16s 436ms/step - accuracy: 0.0000e+00 - loss: 0.0019 -
val_accuracy: 0.0058 - val_loss: 0.0045
Epoch 99/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 11s 463ms/step - accuracy: 0.0000e+00 - loss: 0.0019 -
val_accuracy: 0.0058 - val_loss: 0.0040
Epoch 100/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 21s 496ms/step - accuracy: 0.0000e+00 - loss: 0.0019 -
val_accuracy: 0.0058 - val_loss: 0.0032
```

In [7]:
```
# 6.	Observe the cost function vs iterations learning curve

# Plot cost function (loss) vs iterations (epochs)
plt.figure(figsize=(10, 5))
plt.plot(history.history['loss'], label='Training Loss')
plt.title('Cost Function vs Iterations')
plt.xlabel('Epochs')
plt.ylabel('Loss (MSE)')
plt.legend()
plt.grid(True)
plt.show()
```



Result

In [8]:
```
# a.	Model Summary

model.summary()
```

Model: "sequential"

Layer (type)	Output Shape
lstm (LSTM)	(None, 200, 60)
dropout (Dropout)	(None, 200, 60)
lstm_1 (LSTM)	(None, 200, 60)
dropout_1 (Dropout)	(None, 200, 60)
lstm_2 (LSTM)	(None, 200, 60)
dropout_2 (Dropout)	(None, 200, 60)
lstm_3 (LSTM)	(None, 60)
dropout_3 (Dropout)	(None, 60)
dense (Dense)	(None, 1)

Total params: 306,185 (1.17 MB)

Trainable params: 102,061 (398.68 KB)

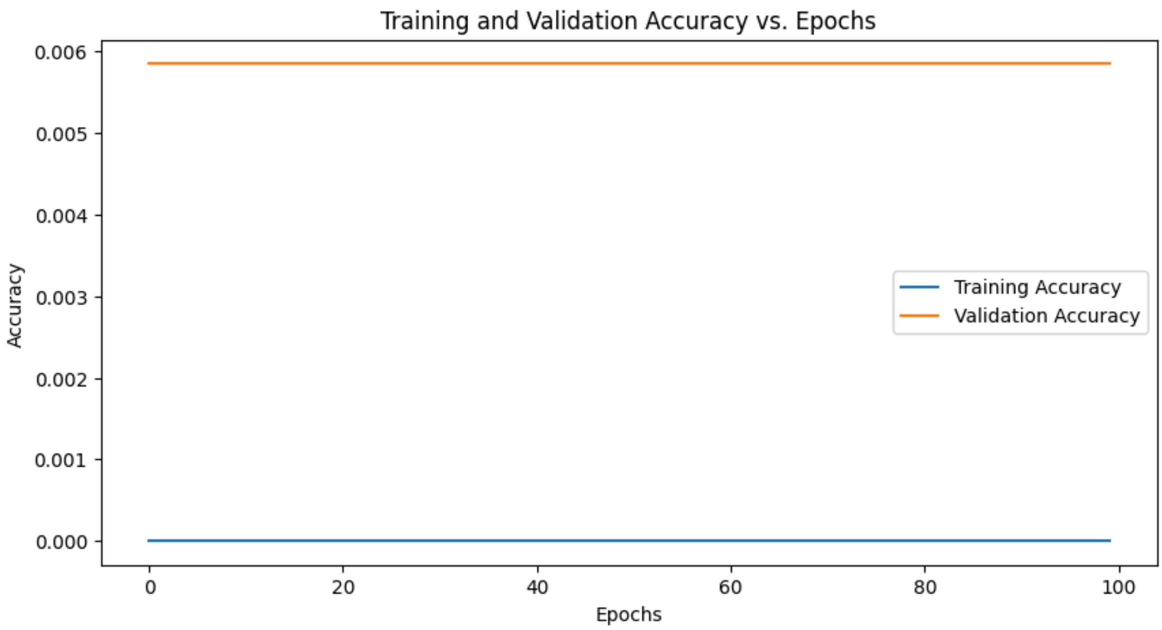
Non-trainable params: 0 (0.00 B)

Optimizer params: 204,124 (797.36 KB)

In [9]:

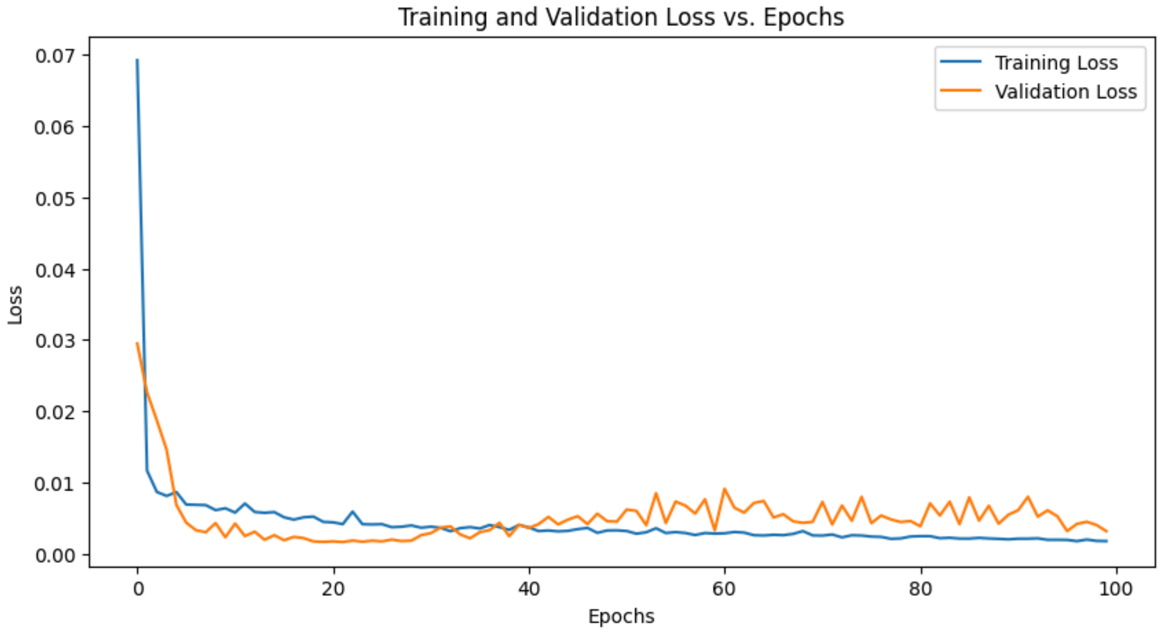
```
# b.    Training and Validation accuracy v/s epochs

# Plot training and validation accuracy
plt.figure(figsize=(10, 5))
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.title('Training and Validation Accuracy vs. Epochs')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



In [10]:
```
# c.    Training and Validation loss v/s epochs

# Plot training and validation loss
plt.figure(figsize=(10, 5))
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.title('Training and Validation Loss vs. Epochs')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```



In [13]:
```
# d. Visualize the Predicted and originalStock Price

# Prepare testing data
testing_data = dataset.iloc[:, 1:2].values
test_feature = testing_data.reshape(-1, 1)
# Convert the 'Price' column to numeric, removing commas
test_feature = test_feature.astype(str)
test_feature = np.char.replace(test_feature, ',', '').astype(float)
test_feature = scaler.transform(test_feature)

testing_features = []

for i in range(60, 253):
    testing_features.append(test_feature[i-60:i, 0])

testing_features = np.array(testing_features)
testing_features = np.reshape(testing_features, (testing_features.shape[0], test

# Predict the stock prices
predictions = model.predict(testing_features)
predictions = scaler.inverse_transform(predictions)

# Visualize the results
plt.figure(figsize=(12, 6))
# Extract the numerical values from testing_data for plotting
plt.plot(test_feature[60:253], color='blue', label='Real TCS Stock Price')
plt.plot(predictions, color='red', label='Predicted TCS Stock Price')
```

```
plt.title('Real vs. Predicted TCS Stock Price')
plt.xlabel('Time')
plt.ylabel('Stock Price')
plt.legend()
plt.show()
```

```
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 35ms/step
```