In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense, Dropout
```

In [2]:
```
data = pd.read_csv('GOOGL.csv')
```

In [3]:
```
scaler = MinMaxScaler(feature_range=(0, 1))
scaled_data = scaler.fit_transform(data['Close'].values.reshape(-1,
```

In [4]:
```
def create_dataset(data, time_step):
    X, y = [], []
    for i in range(len(data) - time_step - 1):
        X.append(data[i:(i + time_step), 0])
        y.append(data[i + time_step, 0])
    return np.array(X), np.array(y)
```

In [5]:
```
time_step = 100
X, y = create_dataset(scaled_data, time_step)
```

In [6]:
```
split_ratio = 0.8
split_index = int(split_ratio * len(data))
X_train, X_test = X[:split_index], X[split_index:]
y_train, y_test = y[:split_index], y[split_index:]
```

In [7]:
```
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)
```

In [8]:
```
model = Sequential()
model.add(LSTM(units=50, return_sequences=True, input_shape=(time_s
model.add(Dropout(0.2))
model.add(LSTM(units=50, return_sequences=True))
model.add(Dropout(0.2))
model.add(LSTM(units=50))
model.add(Dropout(0.2))
model.add(Dense(units=1))
```

In [9]:
```
model.compile(optimizer='adam', loss='mean_squared_error')
```

In [10]:
```
model.fit(X_train, y_train, epochs=5, batch_size=32)
```

```
Epoch 1/5
111/111 [==============================] - 12s 73ms/step - loss:
0.0012
Epoch 2/5
111/111 [==============================] - 8s 76ms/step - loss: 3.
2421e-04
Epoch 3/5
111/111 [==============================] - 8s 76ms/step - loss: 2.
8650e-04
Epoch 4/5
111/111 [==============================] - 8s 76ms/step - loss: 2.
4929e-04
Epoch 5/5
111/111 [==============================] - 8s 76ms/step - loss: 2.
4390e-04
```

Out[10]:
```
<keras.src.callbacks.History at 0x28ffe9650>
```

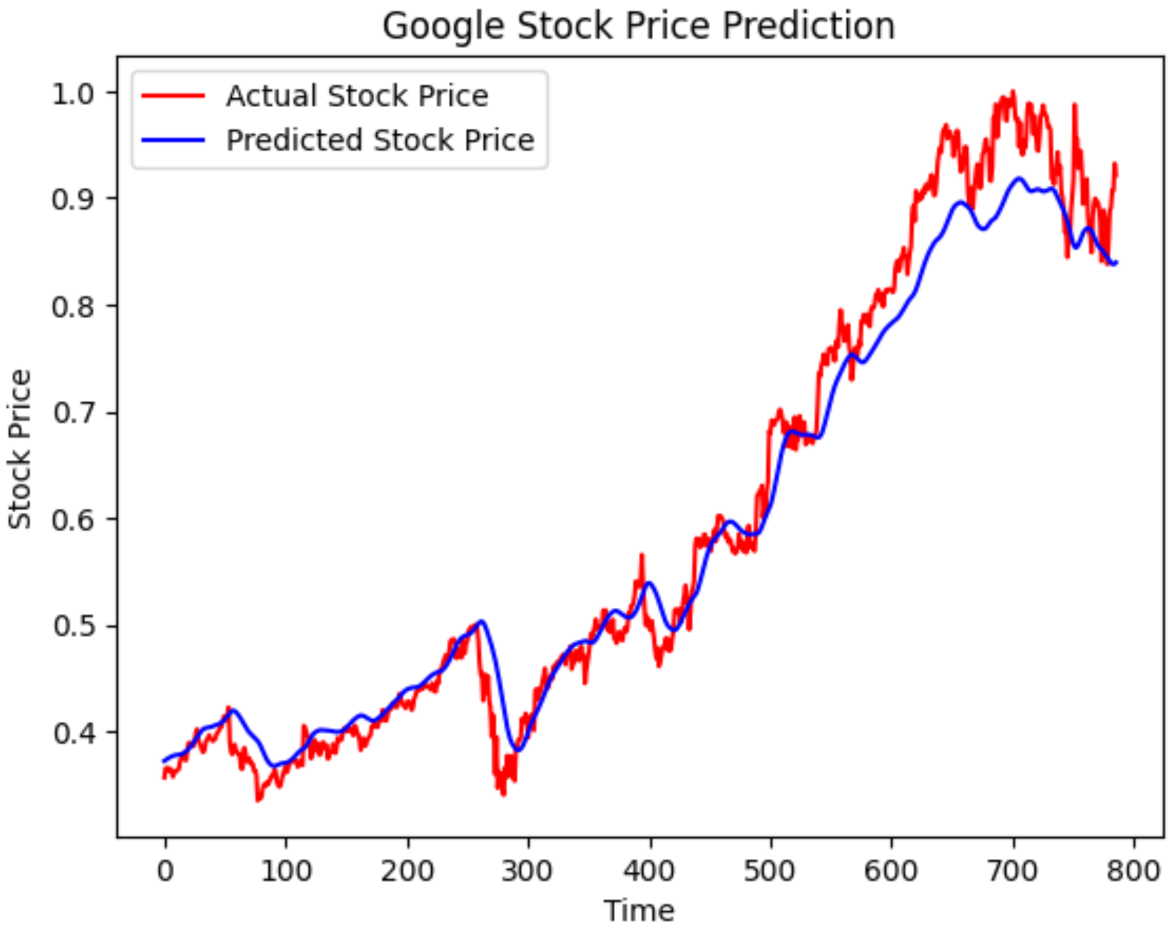
In [11]:
```
loss = model.evaluate(X_test, y_test)
print(f'Test Loss: {loss}')
```

```
25/25 [==============================] - 1s 18ms/step - loss: 0.00
14
Test Loss: 0.0013792210957035422
```

In [12]:
```
predictions = model.predict(X_test)
```

```
25/25 [==============================] - 1s 18ms/step
```

In [13]:
```
plt.plot(y_test, color='red', label='Actual Stock Price')
plt.plot(predictions, color='blue', label='Predicted Stock Price')
plt.title('Google Stock Price Prediction')
plt.xlabel('Time')
plt.ylabel('Stock Price')
plt.legend()
plt.show()
```



In [ ]: