**LSTM LAYER**

In [18]:
```
regressor = Sequential()

regressor.add(LSTM(units = 60, return_sequences = True, input_shape = (X_train.shape[1], 1)))
regressor.add(Dropout(0.1))

regressor.add(LSTM(units = 60, return_sequences = True))
regressor.add(Dropout(0.1))

regressor.add(LSTM(units = 60))
regressor.add(Dropout(0.1))

regressor.add(Dense(units = 1))


regressor.compile(optimizer = 'adam', loss = 'mean_squared_error')
reduce_lr = ReduceLROnPlateau(monitor='val_loss',patience=5)
history =regressor.fit(X_train, Y_train, epochs = 20, batch_size = 15,validation_data=(X_test, Y_test), callbacks=[reduce_lr],shuffle=False)
```

```
Epoch 1/20
212/212 [==============================] - 20s 83ms/step - loss: 0.0046 - val_loss: 0.0264 - lr: 0.0010
Epoch 2/20
212/212 [==============================] - 16s 73ms/step - loss: 0.0119 - val_loss: 0.0496 - lr: 0.0010
Epoch 3/20
212/212 [==============================] - 19s 88ms/step - loss: 0.0119 - val_loss: 0.0572 - lr: 0.0010
Epoch 4/20
212/212 [==============================] - 16s 76ms/step - loss: 0.0154 - val_loss: 0.0508 - lr: 0.0010
Epoch 5/20
212/212 [==============================] - 17s 80ms/step - loss: 0.0178 - val_loss: 0.0457 - lr: 0.0010
Epoch 6/20
212/212 [==============================] - 17s 78ms/step - loss: 0.0194 - val_loss: 0.0540 - lr: 0.0010
Epoch 7/20
212/212 [==============================] - 16s 73ms/step - loss: 0.0290 - val_loss: 0.0038 - lr: 1.0000e-04
Epoch 8/20
212/212 [==============================] - 17s 83ms/step - loss: 0.0035 - val_loss: 0.0029 - lr: 1.0000e-04
```

## MODEL TRAINING

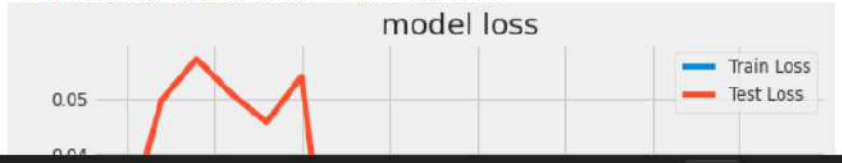
In [19]:
```
train_predict = regressor.predict(X_train)
test_predict = regressor.predict(X_test)
```

In [20]:
```
train_predict = sc.inverse_transform(train_predict)
Y_train = sc.inverse_transform([Y_train])
test_predict = sc.inverse_transform(test_predict)
Y_test = sc.inverse_transform([Y_test])
```

## PREDICTION

In [21]:
```
print('Train Mean Absolute Error:', mean_absolute_error(Y_train[0], train_predict[:,0]))
print('Train Root Mean Squared Error:',np.sqrt(mean_squared_error(Y_train[0], train_predict[:,0])))
print('Test Mean Absolute Error:', mean_absolute_error(Y_test[0], test_predict[:,0]))
print('Test Root Mean Squared Error:',np.sqrt(mean_squared_error(Y_test[0], test_predict[:,0])))
plt.figure(figsize=(8,4))
plt.plot(history.history['loss'], label='Train Loss')
plt.plot(history.history['val_loss'], label='Test Loss')
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epochs')
plt.legend(loc='upper right')
plt.show();
```

```
Train Mean Absolute Error: 3.096717261068476
Train Root Mean Squared Error: 3.8820918567298652
Test Mean Absolute Error: 2.7278705535818837
Test Root Mean Squared Error: 5.479474283362478
```



In [21]:
```
print('Train Mean Absolute Error:', mean_absolute_error(Y_train[0], train_predict[:,0]))
print('Train Root Mean Squared Error:',np.sqrt(mean_squared_error(Y_train[0], train_predict[:,0])))
print('Test Mean Absolute Error:', mean_absolute_error(Y_test[0], test_predict[:,0]))
print('Test Root Mean Squared Error:',np.sqrt(mean_squared_error(Y_test[0], test_predict[:,0])))
plt.figure(figsize=(8,4))
plt.plot(history.history['loss'], label='Train Loss')
plt.plot(history.history['val_loss'], label='Test Loss')
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epochs')
plt.legend(loc='upper right')
plt.show();
```

```
Train Mean Absolute Error: 3.096717261068476
Train Root Mean Squared Error: 3.8820918567298652
Test Mean Absolute Error: 2.7278705535818837
Test Root Mean Squared Error: 5.479474283362478
```