In [75]:
```
# Assignment 5
# Step 1: Import Packages
import numpy as np
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, ReLU, Input
import matplotlib.pyplot as plt
plt.style.use('seaborn-v0_8-whitegrid')
```

In [77]:
```
# Step 2: Data Loading/Processing
# 2.1: Generate synthetic data
np.random.seed(42)
n_samples = 1000
X = np.random.rand(n_samples, 3) * 100 # Features: size, location, bedrooms
y = X[:, 0] * 3 + X[:, 1] * 2 + X[:, 2] * 4 + np.random.randn(n_samples) * 10
```

In [79]:
```
# Split data into train and test sets
from sklearn . model_selection import train_test_split
X_train , X_test , y_train , y_test = train_test_split (X , y ,test_size =0.2 , random_state =42)
```

In [81]:
```
# Define the model
model = Sequential([
    Input(shape=(3,)),  # This replaces input_dim=3
    Dense(64, activation='relu'),
    Dense(32, activation='relu'),
    Dense(1)  # Output layer for regression
])

# Compile the model
model.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate=0.01),
    loss='mean_squared_error',
    metrics=['mae']
)
```

In [83]:
```
# Train the model
history = model . fit (
    X_train , y_train ,
    validation_split =0.2 ,
    epochs =50 ,
    batch_size =32
)
```

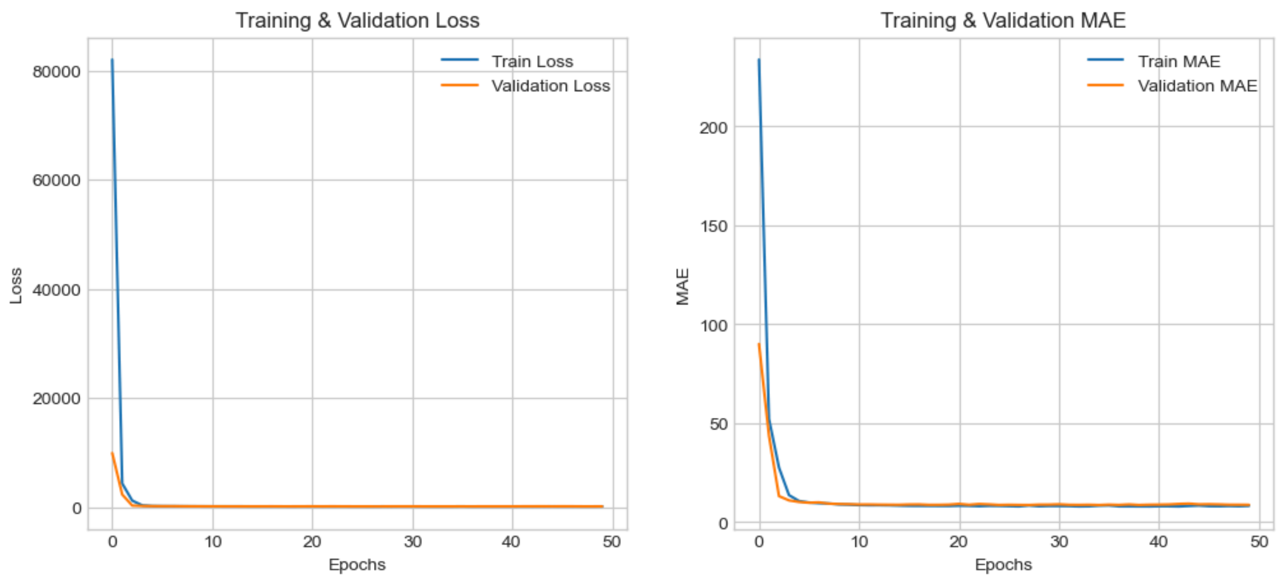
```
Epoch 1/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 1s 6ms/step - loss: 104140.3047 - mae: 273.3638 - val_loss: 2747.4390 - val_mae: 43.8197
Epoch 2/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - loss: 6196.3262 - mae: 66.1622 - val_loss: 1471.7576 - val_mae: 35.6460
Epoch 3/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - loss: 778.7529 - mae: 23.1179 - val_loss: 399.2774 - val_mae: 17.1145
Epoch 4/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - loss: 289.9537 - mae: 13.8677 - val_loss: 185.3257 - val_mae: 11.1586
Epoch 5/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - loss: 154.0318 - mae: 10.0166 - val_loss: 125.8354 - val_mae: 9.1966
Epoch 6/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - loss: 122.6524 - mae: 8.6727 - val_loss: 121.5769 - val_mae: 8.9316
Epoch 7/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - loss: 115.7169 - mae: 8.5081 - val_loss: 121.7193 - val_mae: 8.9457
Epoch 8/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - loss: 110.0613 - mae: 8.5295 - val_loss: 124.0160 - val_mae: 9.0287
Epoch 9/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 105.3692 - mae: 8.1787 - val_loss: 114.1675 - val_mae: 8.8590
Epoch 10/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - loss: 108.2031 - mae: 8.4097 - val_loss: 109.0630 - val_mae: 8.6584
Epoch 11/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - loss: 105.6987 - mae: 8.2669 - val_loss: 118.3935 - val_mae: 8.8721
Epoch 12/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 110.4171 - mae: 8.3680 - val_loss: 111.9170 - val_mae: 8.8390
Epoch 13/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - loss: 96.6011 - mae: 7.9186 - val_loss: 118.5496 - val_mae: 9.0745
Epoch 14/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - loss: 106.3652 - mae: 8.2574 - val_loss: 127.0172 - val_mae: 9.1535
Epoch 15/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 100.5067 - mae: 7.9405 - val_loss: 110.0567 - val_mae: 8.6862
Epoch 16/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 95.2716 - mae: 7.7399 - val_loss: 107.6064 - val_mae: 8.6995
Epoch 17/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 102.8068 - mae: 8.0284 - val_loss: 109.3097 - val_mae: 8.7829
Epoch 18/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 86.3700 - mae: 7.3474 - val_loss: 107.9174 - val_mae: 8.6578
Epoch 19/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 100.6856 - mae: 8.0309 - val_loss: 110.7568 - val_mae: 8.8881
Epoch 20/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 99.8287 - mae: 8.0462 - val_loss: 108.3848 - val_mae: 8.6511
Epoch 21/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 100.4619 - mae: 8.1524 - val_loss: 107.5807 - val_mae: 8.6988
Epoch 22/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 93.3717 - mae: 7.6312 - val_loss: 109.6144 - val_mae: 8.6857
Epoch 23/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 101.7367 - mae: 8.0718 - val_loss: 107.5458 - val_mae: 8.6421
Epoch 24/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 97.8444 - mae: 7.8845 - val_loss: 106.8076 - val_mae: 8.7285
Epoch 25/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 90.1166 - mae: 7.5241 - val_loss: 106.9352 - val_mae: 8.6680
Epoch 26/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - loss: 100.5933 - mae: 8.0134 - val_loss: 129.1834 - val_mae: 9.4099
Epoch 27/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 121.1432 - mae: 8.6934 - val_loss: 120.2903 - val_mae: 9.0318
Epoch 28/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 111.3664 - mae: 8.4048 - val_loss: 119.8266 - val_mae: 8.9881
Epoch 29/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - loss: 103.6499 - mae: 8.1227 - val_loss: 106.0838 - val_mae: 8.6510
Epoch 30/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 112.0898 - mae: 8.4298 - val_loss: 108.1929 - val_mae: 8.6958
Epoch 31/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 104.9371 - mae: 7.9379 - val_loss: 124.0955 - val_mae: 9.0705
Epoch 32/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 112.4431 - mae: 8.5345 - val_loss: 106.4928 - val_mae: 8.7533
Epoch 33/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 103.3258 - mae: 8.1405 - val_loss: 106.4248 - val_mae: 8.6576
Epoch 34/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 99.3586 - mae: 7.7765 - val_loss: 108.4856 - val_mae: 8.8566
Epoch 35/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 105.2988 - mae: 8.1707 - val_loss: 110.3695 - val_mae: 8.6980
Epoch 36/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 104.9999 - mae: 8.2908 - val_loss: 119.0265 - val_mae: 8.9236
Epoch 37/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 103.5646 - mae: 8.0985 - val_loss: 106.9314 - val_mae: 8.6852
Epoch 38/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 95.8234 - mae: 7.8008 - val_loss: 107.1038 - val_mae: 8.7896
Epoch 39/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - loss: 91.6426 - mae: 7.5533 - val_loss: 106.4347 - val_mae: 8.6214
Epoch 40/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 88.9704 - mae: 7.3960 - val_loss: 110.6709 - val_mae: 8.9557
Epoch 41/50
```

```
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 100.7943 - mae: 7.9541 - val_loss: 106.6060 - val_mae: 8.6142
Epoch 42/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 83.1020 - mae: 7.1947 - val_loss: 109.2637 - val_mae: 8.8920
Epoch 43/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 102.1104 - mae: 7.9102 - val_loss: 107.7422 - val_mae: 8.7117
Epoch 44/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - loss: 100.8667 - mae: 7.8962 - val_loss: 106.1636 - val_mae: 8.5890
Epoch 45/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - loss: 100.9298 - mae: 8.0219 - val_loss: 118.4211 - val_mae: 8.9094
Epoch 46/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 109.2541 - mae: 8.5115 - val_loss: 139.4163 - val_mae: 9.5208
Epoch 47/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - loss: 107.6149 - mae: 8.1930 - val_loss: 105.8575 - val_mae: 8.7364
Epoch 48/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 104.3214 - mae: 7.9700 - val_loss: 117.8017 - val_mae: 9.1341
Epoch 49/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - loss: 105.5499 - mae: 8.1785 - val_loss: 110.4186 - val_mae: 8.9046
Epoch 50/50
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - loss: 100.3518 - mae: 7.9029 - val_loss: 105.0513 - val_mae: 8.5977
```

In [64]:
```
# Evaluate on the test set
loss, mae = model.evaluate(X_test, y_test)
print(f"Test Loss: {loss}, Test MAE: {mae}")
```

```
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - loss: 103.7085 - mae: 8.3647
Test Loss: 113.65978240966797, Test MAE: 8.678581237792969
```

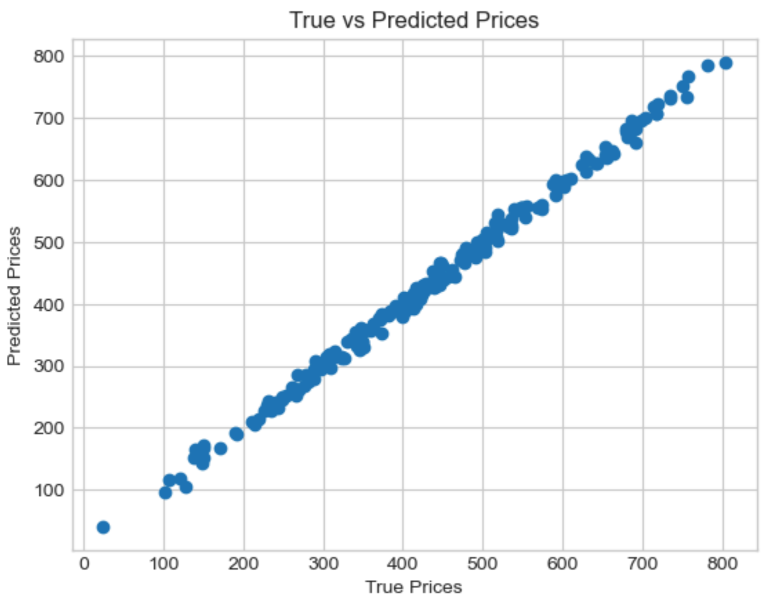
In [65]:
```
# Plot training loss and MAE
fig, ax = plt.subplots(1, 2, figsize=(12, 5))

# Plot loss
ax[0].plot(history.history['loss'], label='Train Loss')
ax[0].plot(history.history['val_loss'], label='Validation Loss')
ax[0].set_xlabel('Epochs')
ax[0].set_ylabel('Loss')
ax[0].set_title('Training & Validation Loss')
ax[0].legend()

# Plot MAE
ax[1].plot(history.history['mae'], label='Train MAE')
ax[1].plot(history.history['val_mae'], label='Validation MAE')
ax[1].set_xlabel('Epochs')
ax[1].set_ylabel('MAE')
ax[1].set_title('Training & Validation MAE')
ax[1].legend()

plt.show()
```



In [66]:
```
# Make predictions
predictions = model . predict ( X_test )
```

```
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step
```

In [67]:
```
plt.scatter(y_test, predictions)
plt.xlabel('True Prices')
plt.ylabel('Predicted Prices')
plt.title('True vs Predicted Prices')
plt.show()
```


True vs Predicted Prices
Predicted Prices
True Prices