In [ ]:
```
pip install pandas numpy matplotlib seaborn scikit-learn plotly streamlit statsmode
```

In [ ]:
```
streamlit run app.py
```

In [55]:
```
# --- IMPORT LIBRARIES ---
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from statsmodels.tsa.seasonal import STL
from datetime import datetime
```

In [21]:
```
# --- LOAD DATA ---

df = pd.read_excel(r"C:\Users\mukki\OneDrive\Desktop\Online Retail.xlsx", sheet_nam
df
```

Out[21]:

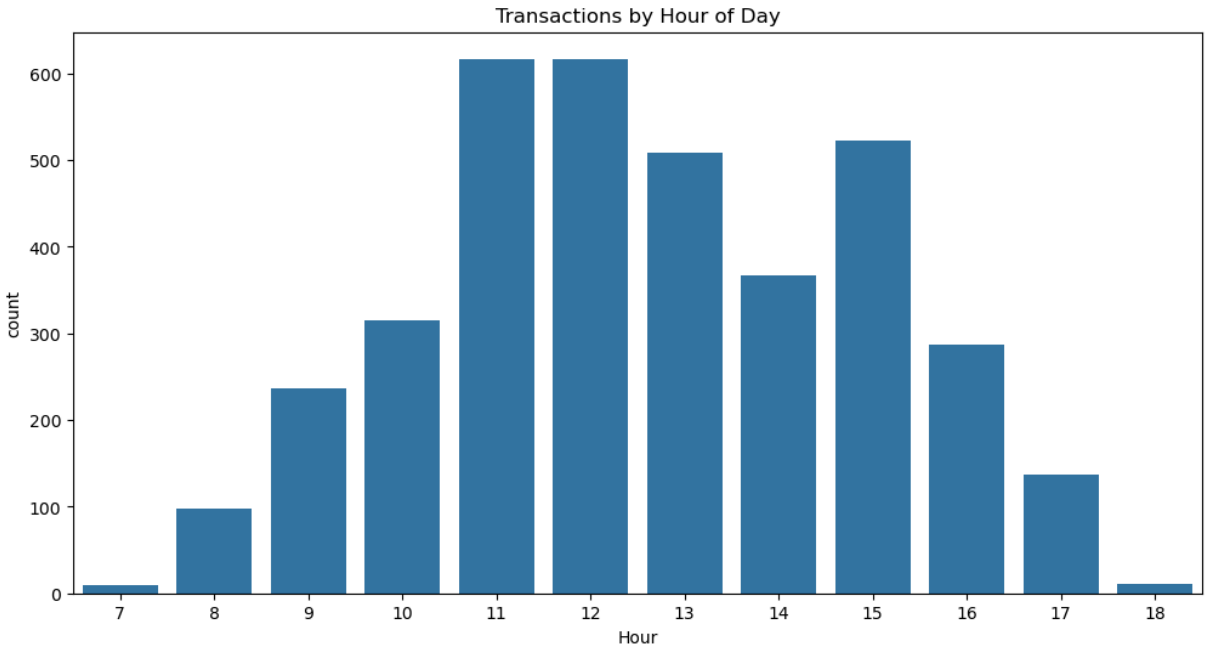
	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	C
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0	K
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0	K
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0	K
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850.0	K
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850.0	K
...	...	...	...	...	...	...	...	
4995	536836	21843	RED RETROSPOT CAKE STAND	2	2010-12-02 18:08:00	10.95	18168.0	K
4996	536836	21531	RED RETROSPOT SUGAR JAM BOWL	2	2010-12-02 18:08:00	2.55	18168.0	K
4997	536836	21539	RED RETROSPOT BUTTER DISH	3	2010-12-02 18:08:00	4.95	18168.0	K
4998	536836	22198	LARGE POPCORN HOLDER	2	2010-12-02 18:08:00	1.65	18168.0	K
4999	536836	22197	SMALL POPCORN HOLDER	2	2010-12-02 18:08:00	0.85	18168.0	K

5000 rows × 8 columns

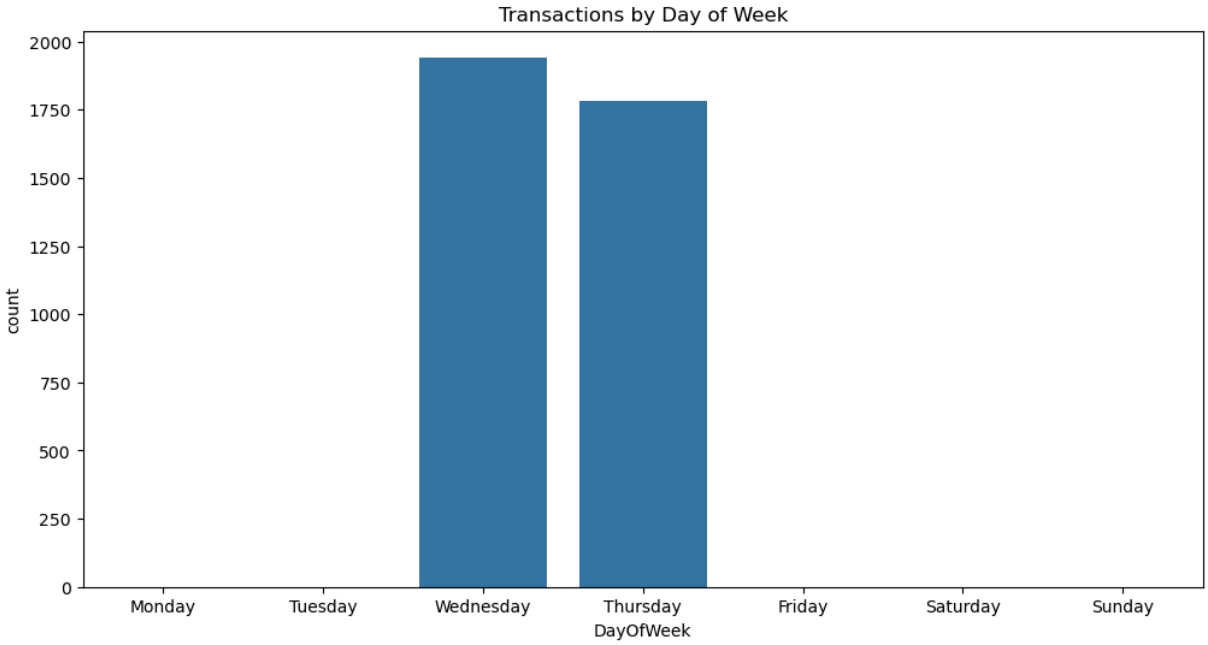
In [25]:
```
# --- BASIC CLEANING ---
df.dropna(subset=['CustomerID'], inplace=True)  # Remove rows with missing customer
df = df[df['Quantity'] > 0]  # Remove canceled/returned items
df = df[df['UnitPrice'] > 0]  # Remove erroneous pricing
```

In [27]:
```
# --- FEATURE ENGINEERING ---
df['TotalPrice'] = df['Quantity'] * df['UnitPrice']
df['InvoiceDate'] = pd.to_datetime(df['InvoiceDate'])
df['Hour'] = df['InvoiceDate'].dt.hour
df['DayOfWeek'] = df['InvoiceDate'].dt.day_name()
df['Month'] = df['InvoiceDate'].dt.month
df['Year'] = df['InvoiceDate'].dt.year
df['Date'] = df['InvoiceDate'].dt.date
```

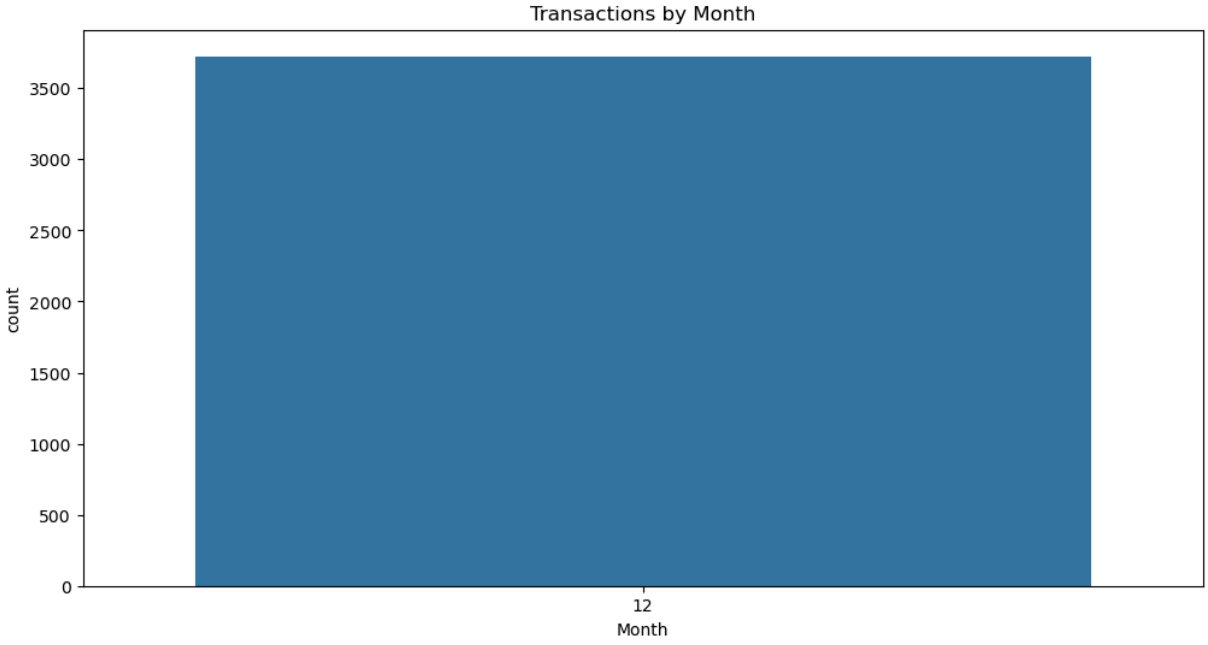
In [29]:
```
# --- TIME-BASED PURCHASING PATTERNS ---
plt.figure(figsize=(12, 6))
sns.countplot(data=df, x='Hour')
plt.title("Transactions by Hour of Day")
plt.show()
```



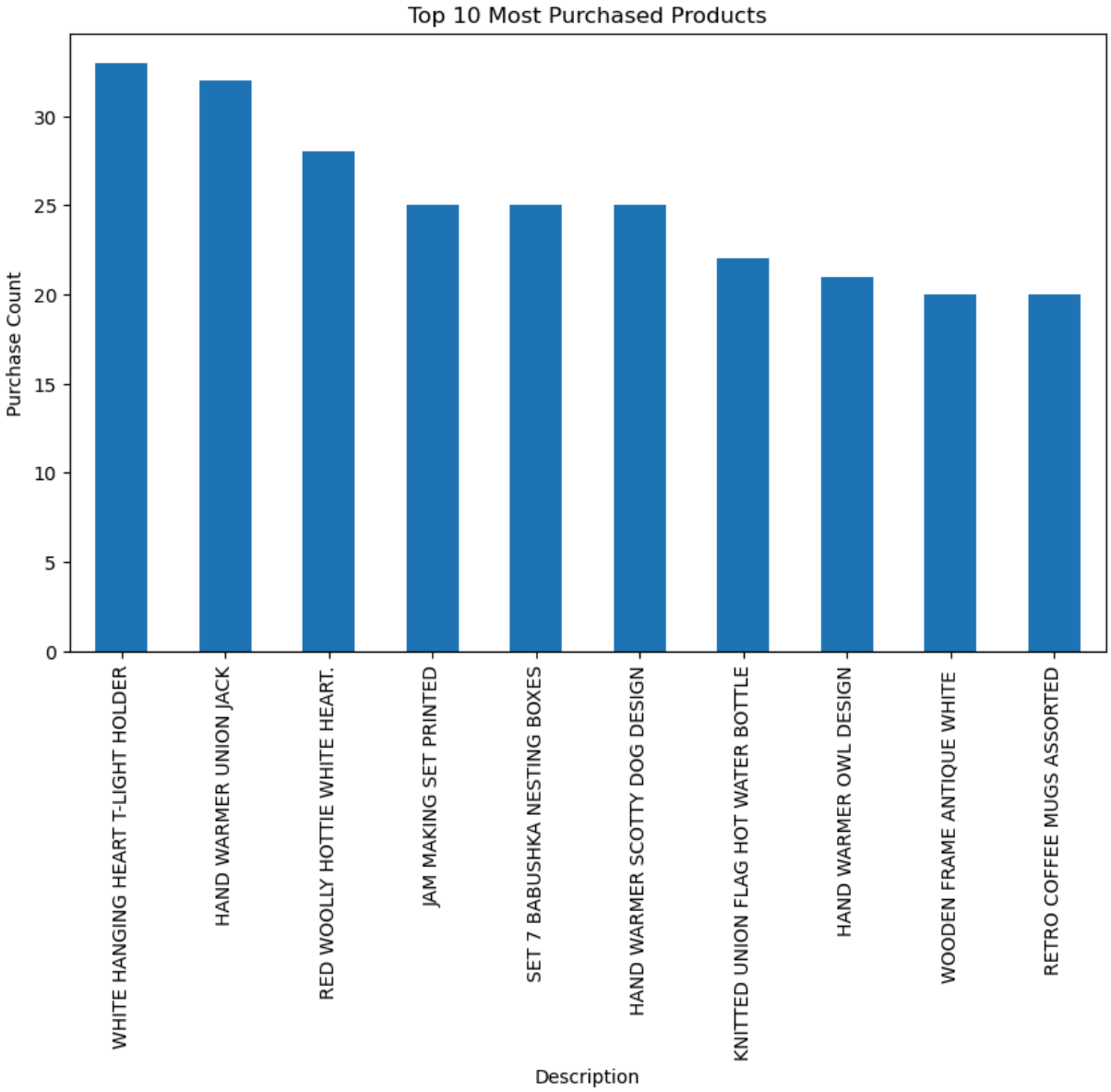
In [31]:
```
plt.figure(figsize=(12, 6))
sns.countplot(data=df, x='DayOfWeek', order=['Monday','Tuesday','Wednesday','Thursd
plt.title("Transactions by Day of Week")
plt.show()
```



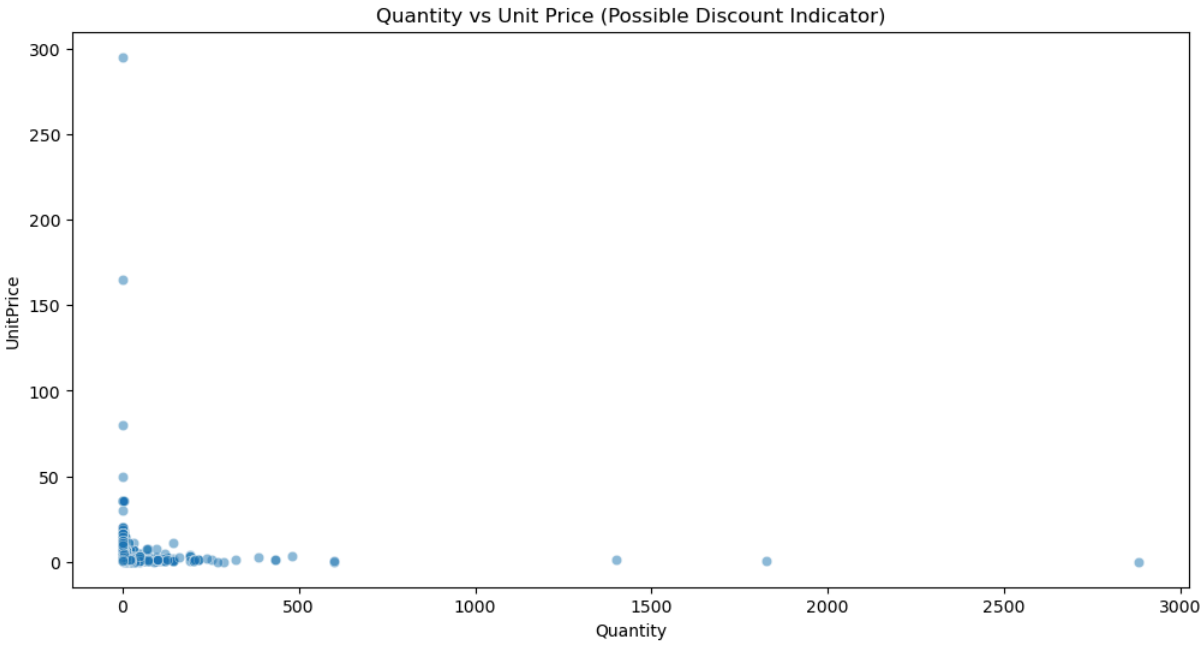
In [33]:
```
plt.figure(figsize=(12, 6))
sns.countplot(data=df, x='Month')
plt.title("Transactions by Month")
plt.show()
```



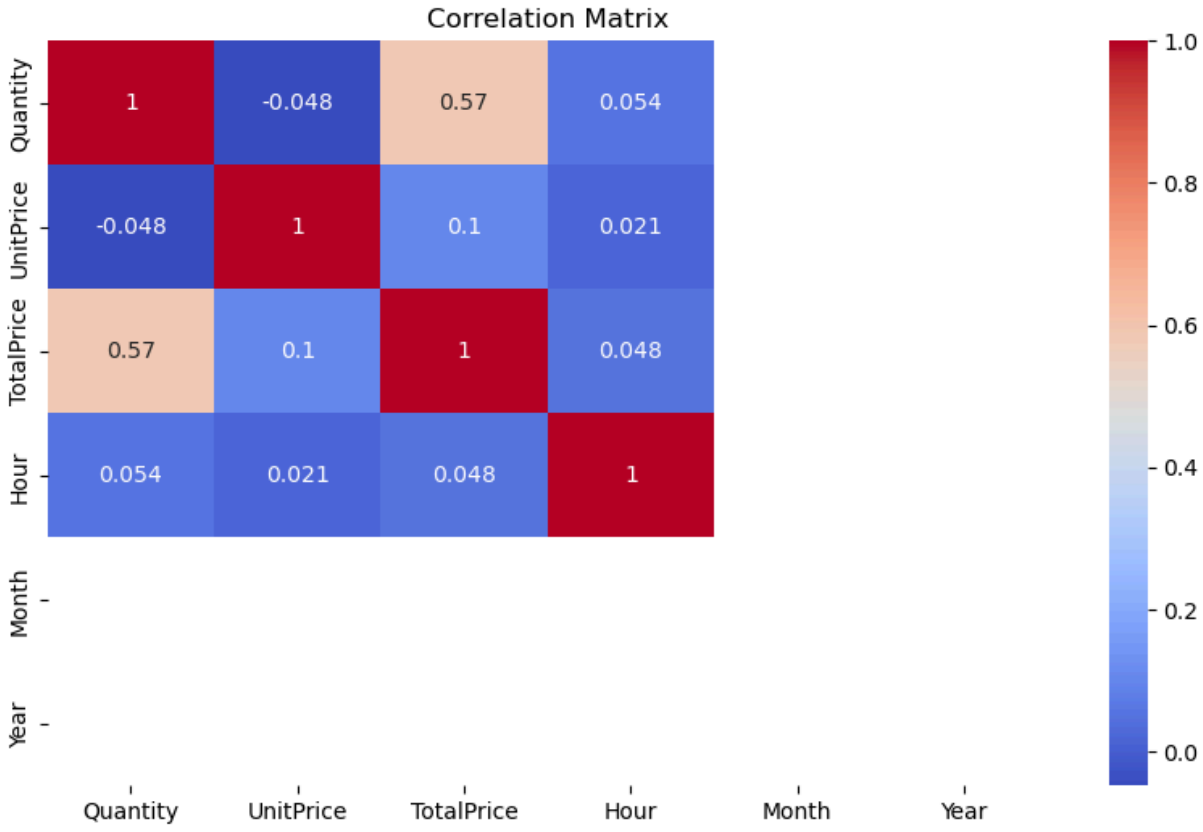
In [35]:
```
# --- TOP PRODUCT CATEGORIES ---
top_products = df['Description'].value_counts().head(10)
plt.figure(figsize=(10, 6))
top_products.plot(kind='bar')
plt.title("Top 10 Most Purchased Products")
plt.ylabel("Purchase Count")
plt.show()
```



In [37]:
```
# --- DISCOUNT ANALYSIS ---
# Assumption: Discounts are reflected in low prices for popular products
plt.figure(figsize=(12, 6))
sns.scatterplot(data=df, x='Quantity', y='UnitPrice', alpha=0.5)
plt.title("Quantity vs Unit Price (Possible Discount Indicator)")
plt.show()
```



In [39]:
```
# --- CORRELATION MATRIX ---
corr = df[['Quantity', 'UnitPrice', 'TotalPrice', 'Hour', 'Month', 'Year']].corr()
plt.figure(figsize=(10, 6))
sns.heatmap(corr, annot=True, cmap='coolwarm')
plt.title("Correlation Matrix")
plt.show()
```

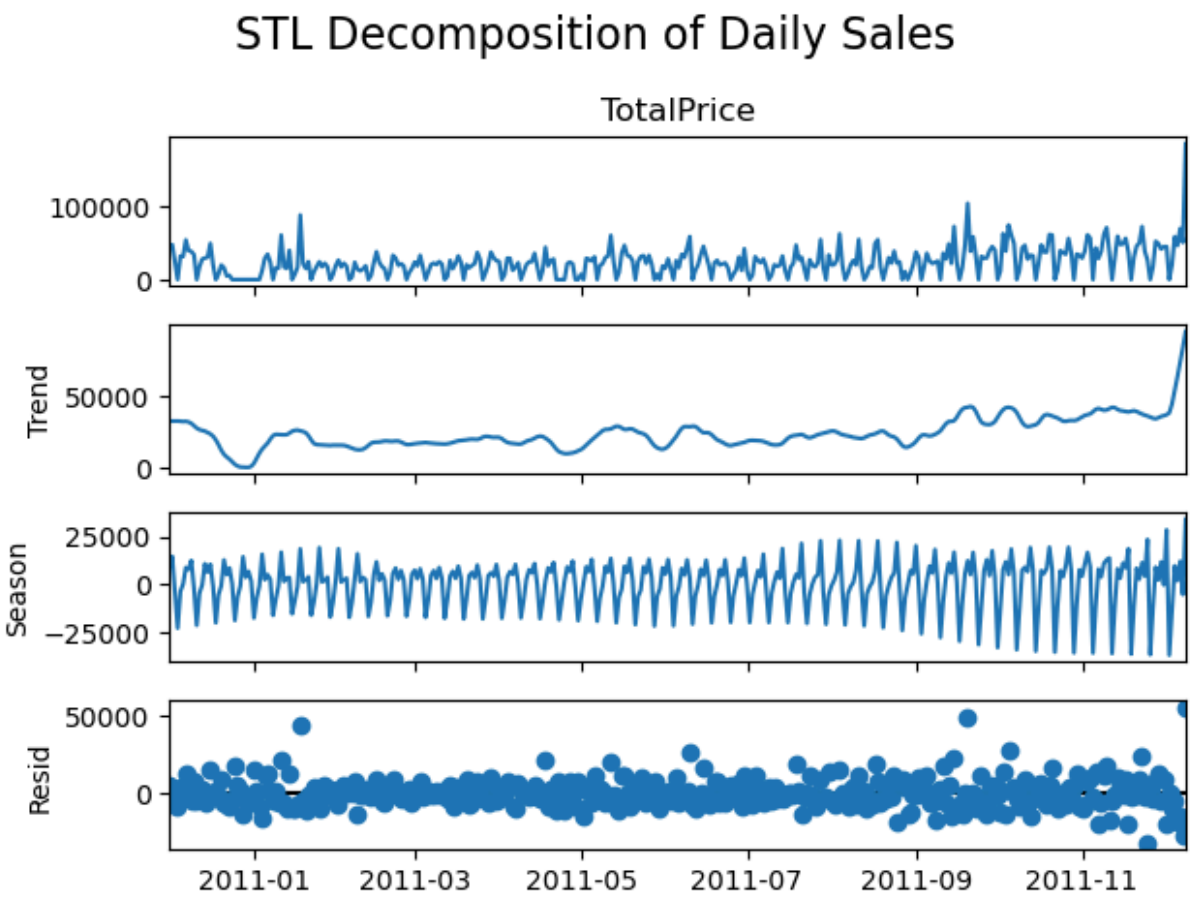


In [53]:
```
# --- SEASONALITY ANALYSIS USING STL ---
# Aggregate daily sales
daily_sales = df.groupby('Date')['TotalPrice'].sum()
```

```
daily_sales.index = pd.to_datetime(daily_sales.index)
daily_sales = daily_sales.asfreq('D').fillna(0)
```

In [49]:
```
plt.figure(figsize=(12, 8))
result.plot()
plt.suptitle("STL Decomposition of Daily Sales", fontsize=16)
plt.tight_layout()
plt.show()
```

```
<Figure size 1200x800 with 0 Axes>
```



In [ ]: