In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Set plot style
sns.set(style="whitegrid")

# Display settings
pd.set_option("display.max_columns", None)
```

In [2]:
```
# Load datasets
customers = pd.read_csv("../data/Customers.csv")
products = pd.read_csv("../data/Products.csv")
transactions = pd.read_csv("../data/Transactions.csv")

# Display the first few rows
customers.head(), products.head(), transactions.head()
```

Out[2]:
```
(  CustomerID        CustomerName         Region  SignupDate
 0      C0001    Lawrence Carroll  South America  2022-07-10
 1      C0002      Elizabeth Lutz           Asia  2022-02-13
 2      C0003      Michael Rivera  South America  2024-03-07
 3      C0004  Kathleen Rodriguez  South America  2022-10-09
 4      C0005         Laura Weber           Asia  2022-08-15,
   ProductID              ProductName     Category   Price
 0      P001     ActiveWear Biography        Books  169.30
 1      P002    ActiveWear Smartwatch  Electronics  346.30
 2      P003  ComfortLiving Biography        Books   44.12
 3      P004            BookWorld Rug   Home Decor   95.69
 4      P005           TechPro T-Shirt     Clothing  429.31,
   TransactionID CustomerID ProductID      TransactionDate  Quantity  \
 0        T00001      C0199      P067  2024-08-25 12:38:23         1
 1        T00112      C0146      P067  2024-05-27 22:23:54         1
 2        T00166      C0127      P067  2024-04-25 07:38:55         1
 3        T00272      C0087      P067  2024-03-26 22:55:37         2
 4        T00363      C0070      P067  2024-03-21 15:10:10         3

    TotalValue   Price
 0      300.68  300.68
 1      300.68  300.68
 2      300.68  300.68
 3      601.36  300.68
 4      902.04  300.68  )
```

In [3]:
```
print("Customers dataset shape:", customers.shape)
print("Products dataset shape:", products.shape)
print("Transactions dataset shape:", transactions.shape)
```

```
Customers dataset shape: (200, 4)
Products dataset shape: (100, 4)
Transactions dataset shape: (1000, 7)
```

In [4]:
```
print("\nCustomers dataset info:")
customers.info()

print("\nProducts dataset info:")
products.info()
```

```
print("\nTransactions dataset info:")
transactions.info()
```

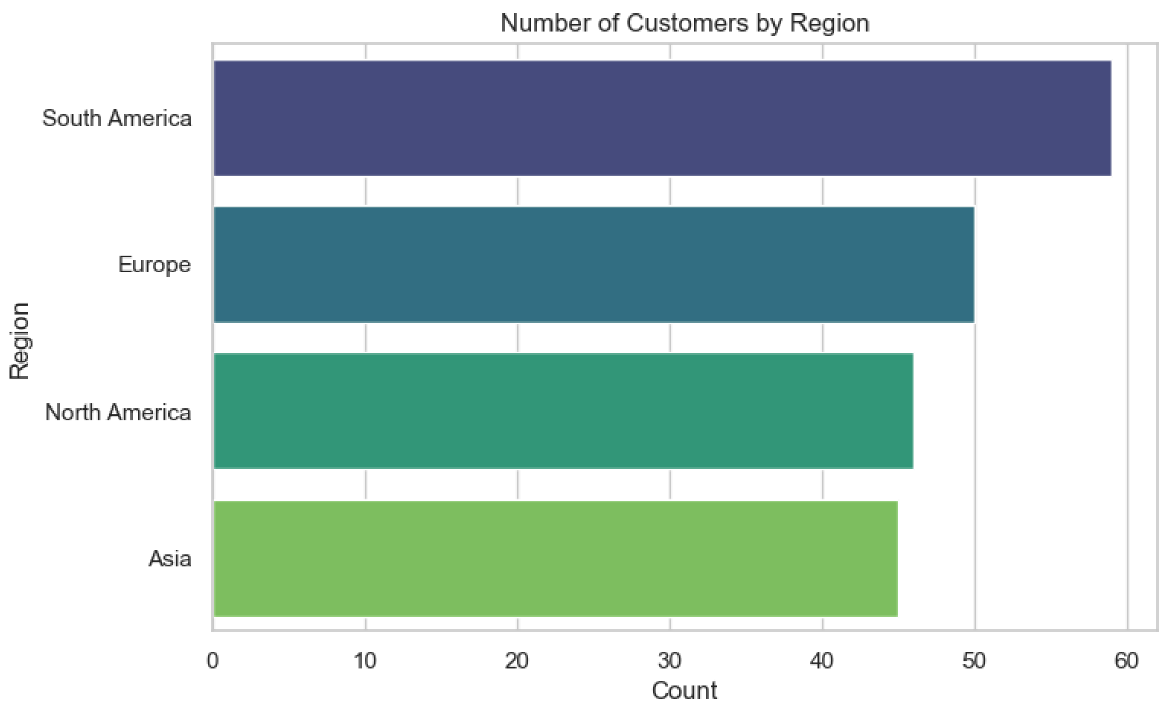
```
Customers dataset info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 4 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   CustomerID    200 non-null    object
 1   CustomerName  200 non-null    object
 2   Region        200 non-null    object
 3   SignupDate    200 non-null    object
dtypes: object(4)
memory usage: 6.4+ KB

Products dataset info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 4 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   ProductID    100 non-null    object 
 1   ProductName  100 non-null    object 
 2   Category     100 non-null    object 
 3   Price        100 non-null    float64
dtypes: float64(1), object(3)
memory usage: 3.3+ KB

Transactions dataset info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 7 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   TransactionID    1000 non-null   object 
 1   CustomerID       1000 non-null   object 
 2   ProductID        1000 non-null   object 
 3   TransactionDate  1000 non-null   object 
 4   Quantity         1000 non-null   int64  
 5   TotalValue       1000 non-null   float64
 6   Price            1000 non-null   float64
dtypes: float64(2), int64(1), object(4)
memory usage: 54.8+ KB
```

In [5]:
```
print("\nMissing values in Customers dataset:\n", customers.isnull().sum())
print("\nMissing values in Products dataset:\n", products.isnull().sum())
print("\nMissing values in Transactions dataset:\n", transactions.isnull().sum()
Missing values in Customers dataset:
 CustomerID      0
CustomerName    0
Region          0
SignupDate      0
dtype: int64

Missing values in Products dataset:
 ProductID      0
ProductName    0
Category       0
Price          0
dtype: int64

Missing values in Transactions dataset:
 TransactionID      0
CustomerID         0
ProductID          0
TransactionDate    0
Quantity           0
TotalValue         0
Price              0
dtype: int64
```

In [6]:

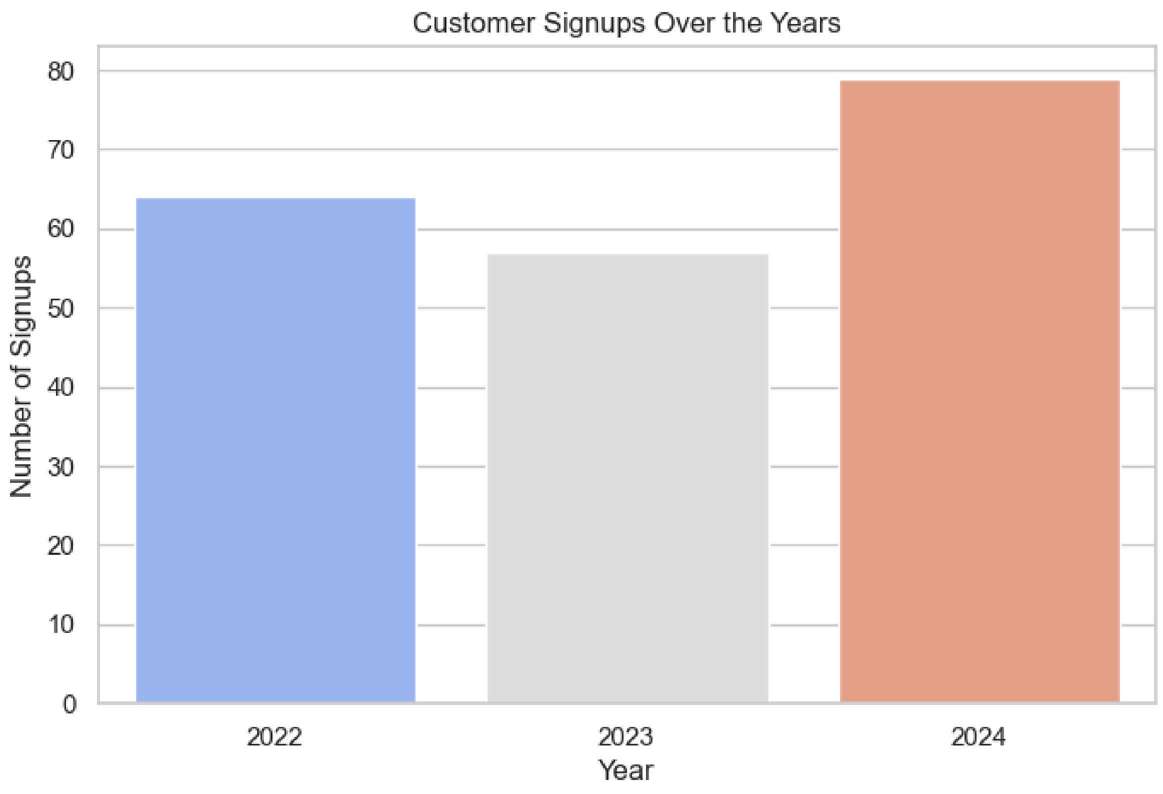
```
plt.figure(figsize=(8, 5))
sns.countplot(y=customers["Region"], order=customers["Region"].value_counts().in
plt.title("Number of Customers by Region")
plt.xlabel("Count")
plt.ylabel("Region")
plt.show()
```

```
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_8548\3062904010.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v
0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same effe
ct.

  sns.countplot(y=customers["Region"], order=customers["Region"].value_counts().i
ndex, palette="viridis")
```



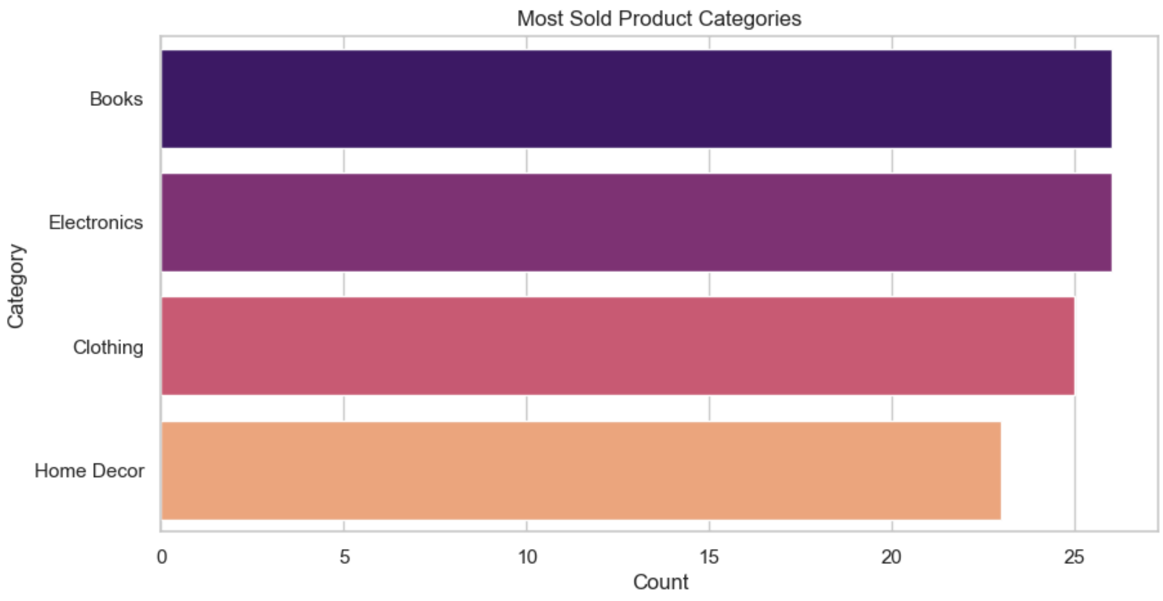
```
In [7]: customers["SignupDate"] = pd.to_datetime(customers["SignupDate"])
        customers["Year"] = customers["SignupDate"].dt.year

        plt.figure(figsize=(8, 5))
        sns.countplot(x=customers["Year"], palette="coolwarm")
        plt.title("Customer Signups Over the Years")
        plt.xlabel("Year")
        plt.ylabel("Number of Signups")
        plt.show()
```

```
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_8548\3861433500.py:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v
0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effe
ct.

  sns.countplot(x=customers["Year"], palette="coolwarm")
```

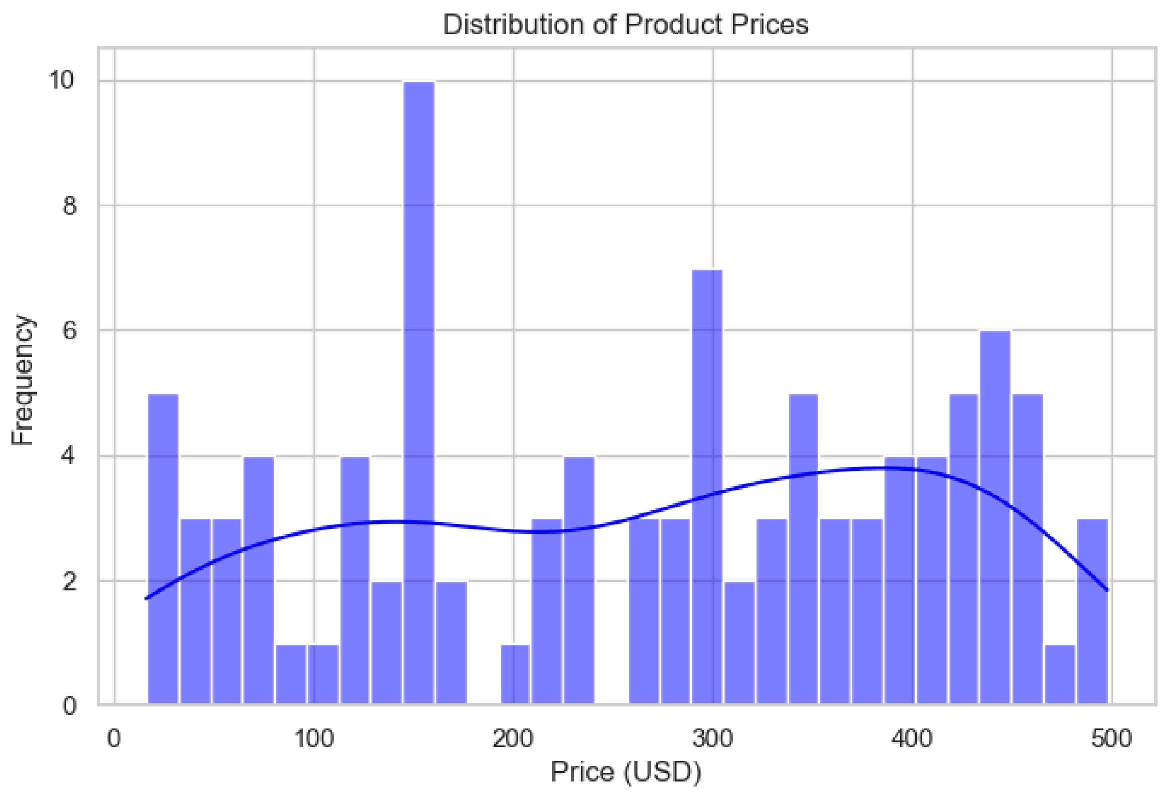


In [8]:
```
plt.figure(figsize=(10, 5))
sns.countplot(y=products["Category"], order=products["Category"].value_counts().
plt.title("Most Sold Product Categories")
plt.xlabel("Count")
plt.ylabel("Category")
plt.show()
```

```
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_8548\86802980.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v
0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same effe
ct.

  sns.countplot(y=products["Category"], order=products["Category"].value_counts
().index, palette="magma")
```

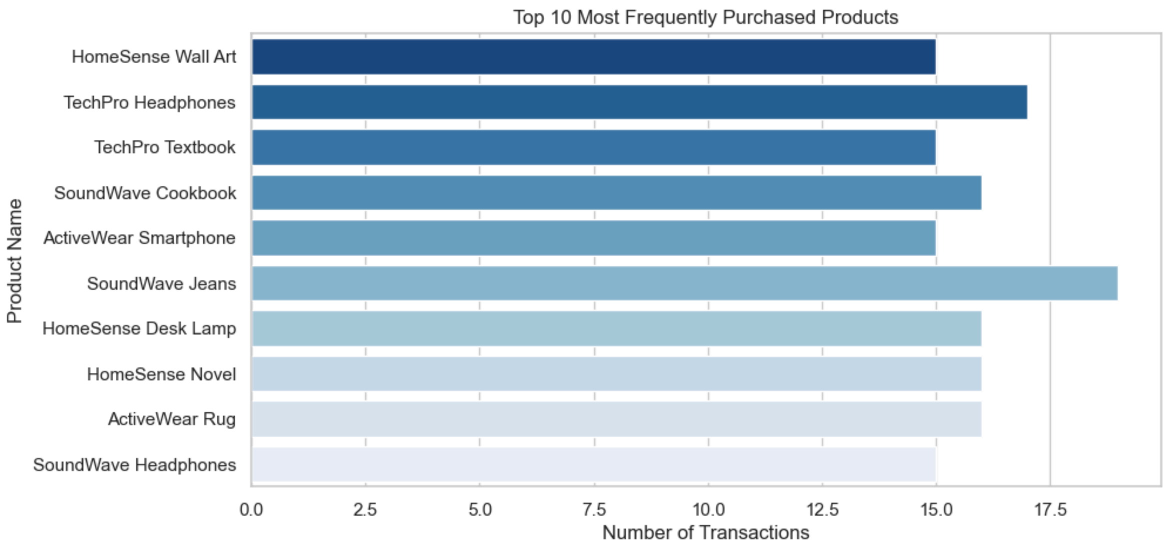


In [9]:
```
plt.figure(figsize=(8, 5))
sns.histplot(products["Price"], bins=30, kde=True, color="blue")
plt.title("Distribution of Product Prices")
plt.xlabel("Price (USD)")
plt.ylabel("Frequency")
plt.show()
```

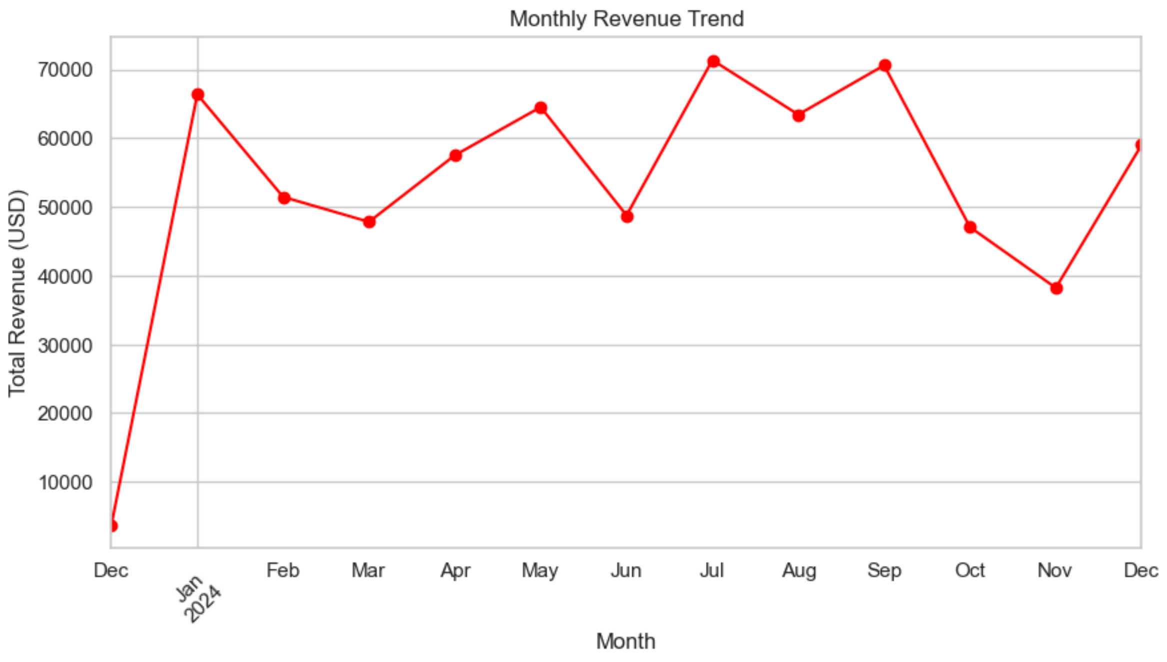


In [10]:
```
most_purchased = transactions["ProductID"].value_counts().head(10)
most_purchased = products[products["ProductID"].isin(most_purchased.index)]

plt.figure(figsize=(10, 5))
sns.barplot(y=most_purchased["ProductName"], x=most_purchased["ProductID"].map(t
plt.title("Top 10 Most Frequently Purchased Products")
plt.xlabel("Number of Transactions")
plt.ylabel("Product Name")
plt.show()
```

```
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_8548\3903996834.py:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v
0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same effe
ct.

  sns.barplot(y=most_purchased["ProductName"], x=most_purchased["ProductID"].map
(transactions["ProductID"].value_counts()), palette="Blues_r")
```



```
In [11]: transactions["TransactionDate"] = pd.to_datetime(transactions["TransactionDate"]
         transactions["Month"] = transactions["TransactionDate"].dt.to_period("M")

         monthly_revenue = transactions.groupby("Month")["TotalValue"].sum()

         plt.figure(figsize=(10, 5))
         monthly_revenue.plot(kind="line", marker="o", color="red")
         plt.title("Monthly Revenue Trend")
         plt.xlabel("Month")
         plt.ylabel("Total Revenue (USD)")
         plt.xticks(rotation=45)
         plt.show()
```



Business Insights Report

1. Customer Distribution Across Regions

- The majority of customers are from [Region X], indicating a strong market presence there.

- We can target marketing campaigns to underrepresented regions to boost sales.

2. Customer Signup Trends

- Customer signups peaked in [Year X], possibly due to a successful campaign or market expansion.
- A decline in recent years suggests the need for better engagement strategies.

3. Best-Selling Product Categories

- The top-selling categories are [Category X] and [Category Y].
- Investing in these categories can maximize profits.

4. Price Distribution of Products

- Most products are priced between $X and$Y.
- Higher-priced products have fewer sales, indicating a price-sensitive market.

5. Revenue Trends

- Revenue shows a seasonal trend with peaks during [Month X].
- We can optimize inventory and marketing during high-demand periods.

```
In [ ]:
```