In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import matplotlib.pyplot as plt
```

In [3]:
```
import warnings
warnings.filterwarnings('ignore')
```

In [8]:
```
df = pd.read_csv("Downloads/supermarket_sales.csv")
df.head()
```

Out[8]:

	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Total	D
0	750-67-8428	A	Yangon	Member	Female	Health and beauty	74.69	7	26.1415	548.9715	1/5/2
1	226-31-3081	C	Naypyitaw	Normal	Female	Electronic accessories	15.28	5	3.8200	80.2200	3/8/2
2	631-41-3108	A	Yangon	Normal	Male	Home and lifestyle	46.33	7	16.2155	340.5255	3/3/2
3	123-19-1176	A	Yangon	Member	Male	Health and beauty	58.22	8	23.2880	489.0480	1/27/2
4	373-73-7910	A	Yangon	Normal	Male	Sports and travel	86.31	7	30.2085	634.3785	2/8/2

In [9]:
```
df.shape
```

Out[9]:
```
(1000, 17)
```

In [10]:
```
df.tail()
```

Out[10]:

	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Total	
995	233-67-5758	C	Naypyitaw	Normal	Male	Health and beauty	40.35	1	2.0175	42.3675	1/
996	303-96-2227	B	Mandalay	Normal	Female	Home and lifestyle	97.38	10	48.6900	1022.4900	3
997	727-02-1313	A	Yangon	Member	Male	Food and beverages	31.84	1	1.5920	33.4320	2

	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Total	
998	347-56-2442	A	Yangon	Normal	Male	Home and lifestyle	65.82	1	3.2910	69.1110	2/
999	849-09-3807	A	Yangon	Member	Female	Fashion accessories	88.34	7	30.9190	649.2990	2/

In [11]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 17 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   Invoice ID               1000 non-null   object 
 1   Branch                   1000 non-null   object 
 2   City                     1000 non-null   object 
 3   Customer type            1000 non-null   object 
 4   Gender                   1000 non-null   object 
 5   Product line             1000 non-null   object 
 6   Unit price               1000 non-null   float64
 7   Quantity                 1000 non-null   int64  
 8   Tax 5%                   1000 non-null   float64
 9   Total                    1000 non-null   float64
 10  Date                     1000 non-null   object 
 11  Time                     1000 non-null   object 
 12  Payment                  1000 non-null   object 
 13  cogs                     1000 non-null   float64
 14  gross margin percentage  1000 non-null   float64
 15  gross income             1000 non-null   float64
 16  Rating                   1000 non-null   float64
dtypes: float64(7), int64(1), object(9)
memory usage: 132.9+ KB
```

In [12]:
```
df.sample()
```

Out[12]:

	Invoice ID	Branch	City	Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Total	D
690	394-55-6384	C	Naypyitaw	Member	Female	Sports and travel	70.19	9	31.5855	663.2955	1/25/2

In [13]:
```
#check for NULL values
df.isnull().sum()
```

Out[13]:
```
Invoice ID                0
Branch                    0
City                      0
Customer type             0
Gender                    0
Product line              0
Unit price                0
```

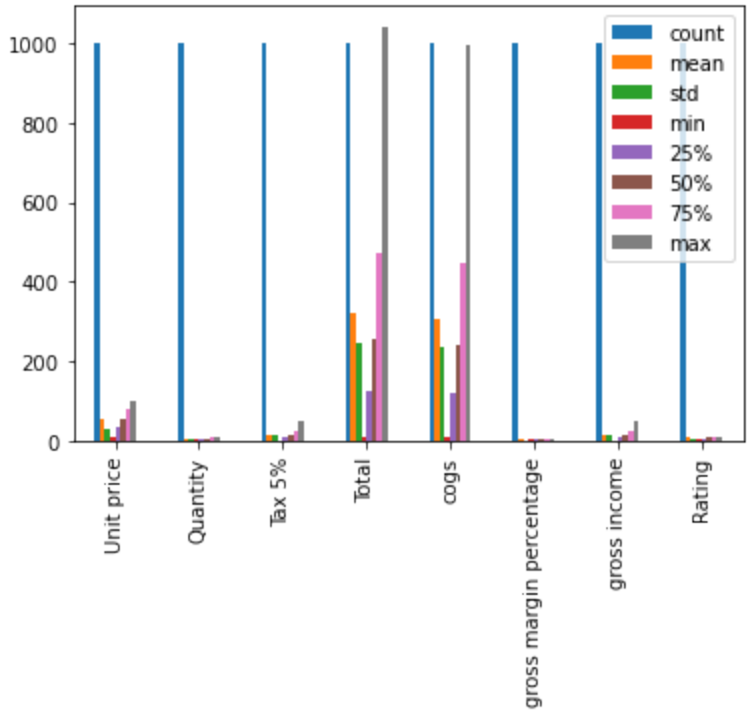
```
Quantity                  0
Tax 5%                    0
Total                     0
Date                      0
Time                      0
Payment                   0
cogs                      0
gross margin percentage   0
gross income              0
Rating                    0
dtype: int64
```

In [14]:
```
#check for Duplicate rows
df.duplicated().sum()
```

Out[14]:
```
0
```

In [15]:
```
df.describe().T.plot(kind='bar')
```

Out[15]:
```
<AxesSubplot:>
```

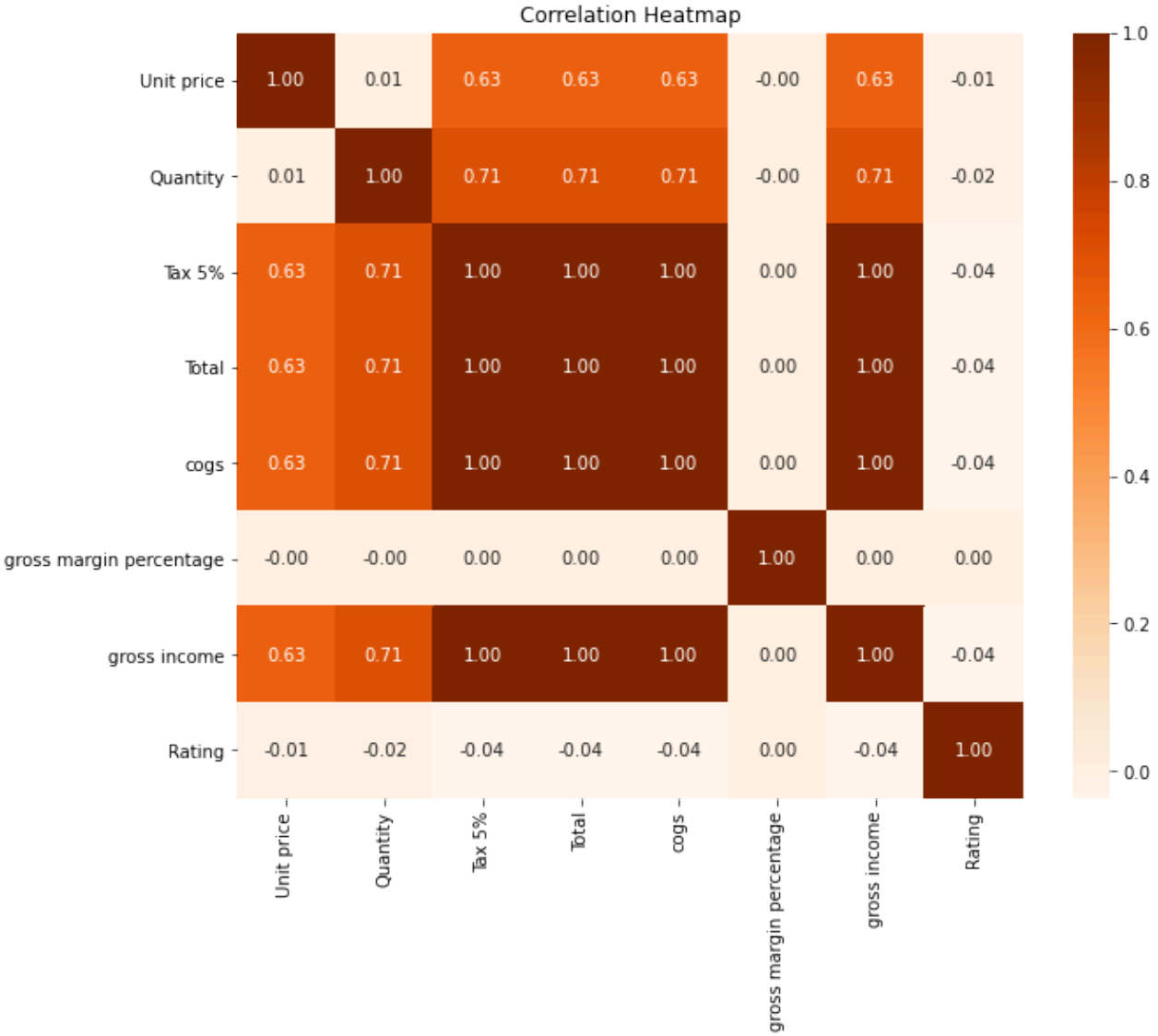


In [29]:
```
# Identify categorical columns
categorical_columns = df.select_dtypes(include=['object', 'category']).columns
print("Categorical Columns:")
print(categorical_columns)

# Identify numerical columns
numerical_columns = df.select_dtypes(include=['number']).columns
print("\nNumerical Columns:")
print(numerical_columns)
```

```
Categorical Columns:
Index(['Invoice ID', 'Branch', 'City', 'Customer type', 'Gender',
       'Product line', 'Date', 'Time', 'Payment'],
      dtype='object')

Numerical Columns:
Index(['Unit price', 'Quantity', 'Tax 5%', 'Total', 'cogs',
```

```
       'gross margin percentage', 'gross income', 'Rating'],
      dtype='object')
```

In [16]:
```
#Select numeric columns
numeric_df = df.select_dtypes(include=[np.number])

#Correlation heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(numeric_df.corr(), annot=True, cmap='Oranges', fmt='.2f')
plt.title('Correlation Heatmap')
plt.show()
```



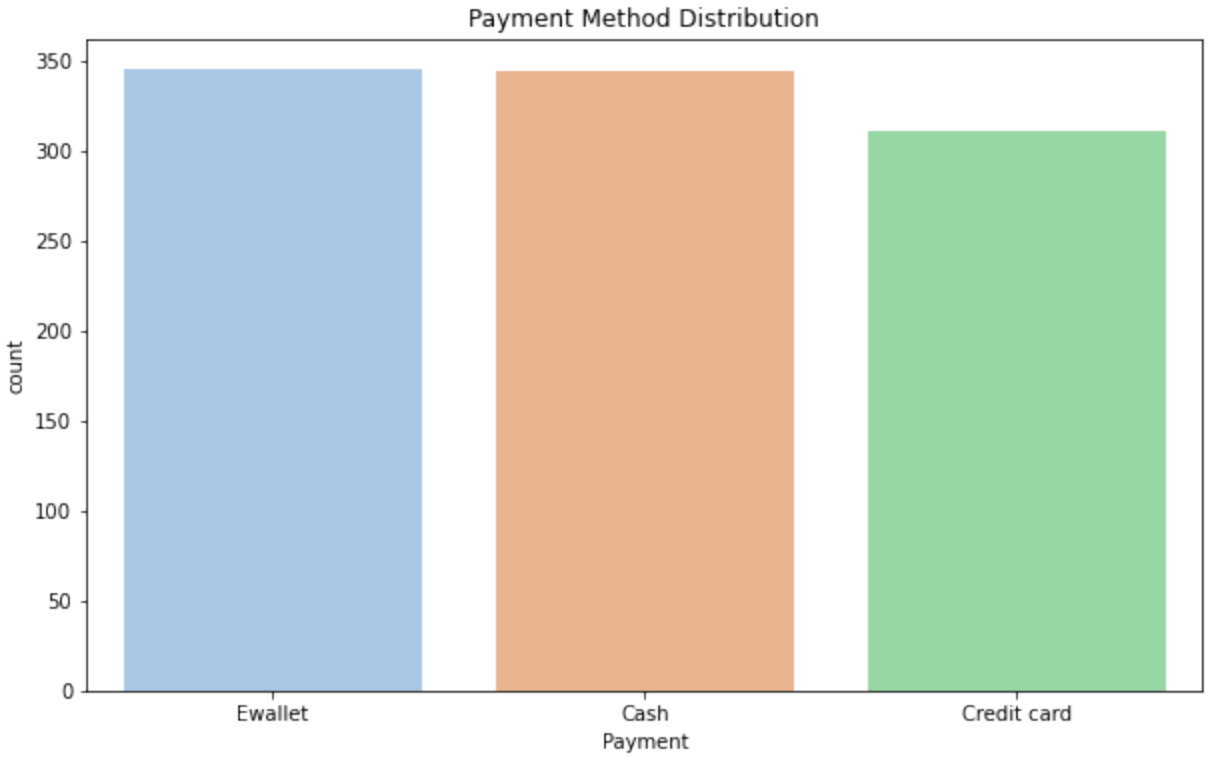
In [23]:
```
df.columns.to_list()
```

Out[23]:
```
['Invoice ID',
 'Branch',
 'City',
 'Customer type',
 'Gender',
 'Product line',
 'Unit price',
 'Quantity',
 'Tax 5%',
 'Total',
 'Date',
```

```
'Time',
'Payment',
'cogs',
'gross margin percentage',
'gross income',
'Rating']
```

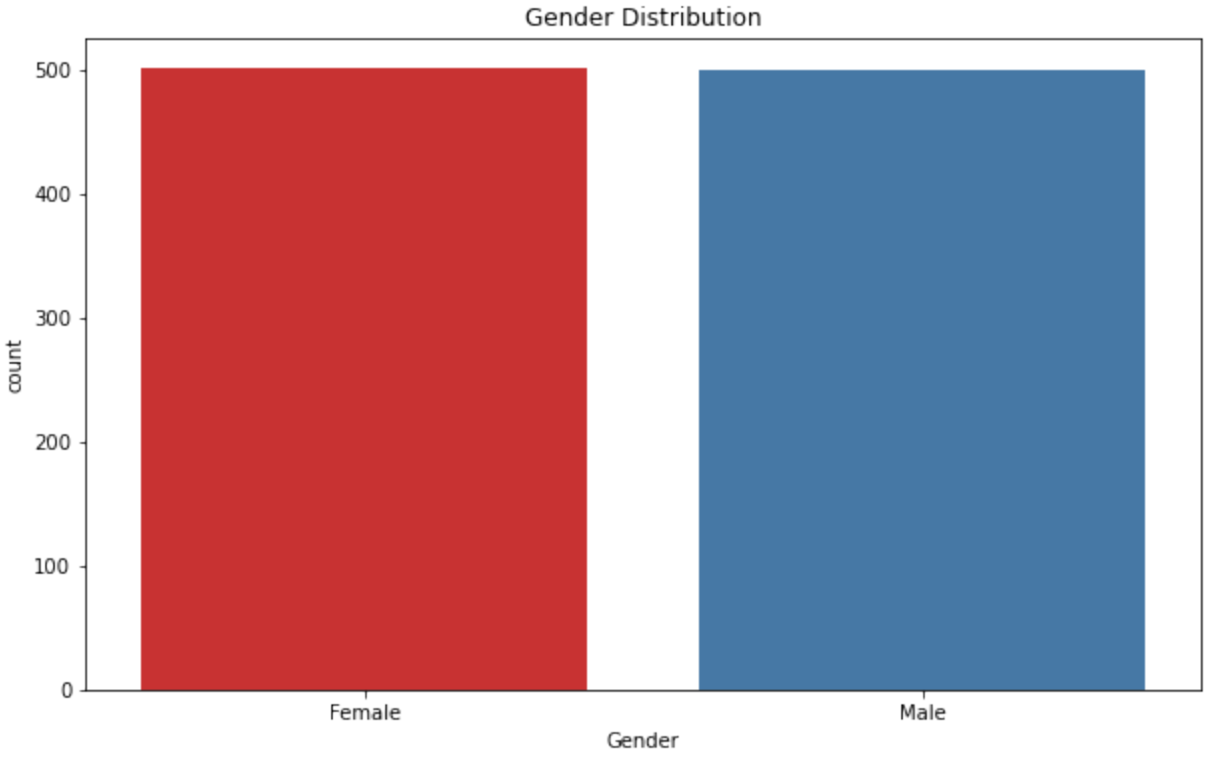
Sales Distribution by Payment Method Used

In [33]:
```
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x='Payment', palette='pastel')
plt.title('Payment Method Distribution')
plt.show()
```



Sales Distribution by Gender

In [37]:
```
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x='Gender', palette='Set1')
plt.title('Gender Distribution')
plt.show()
```

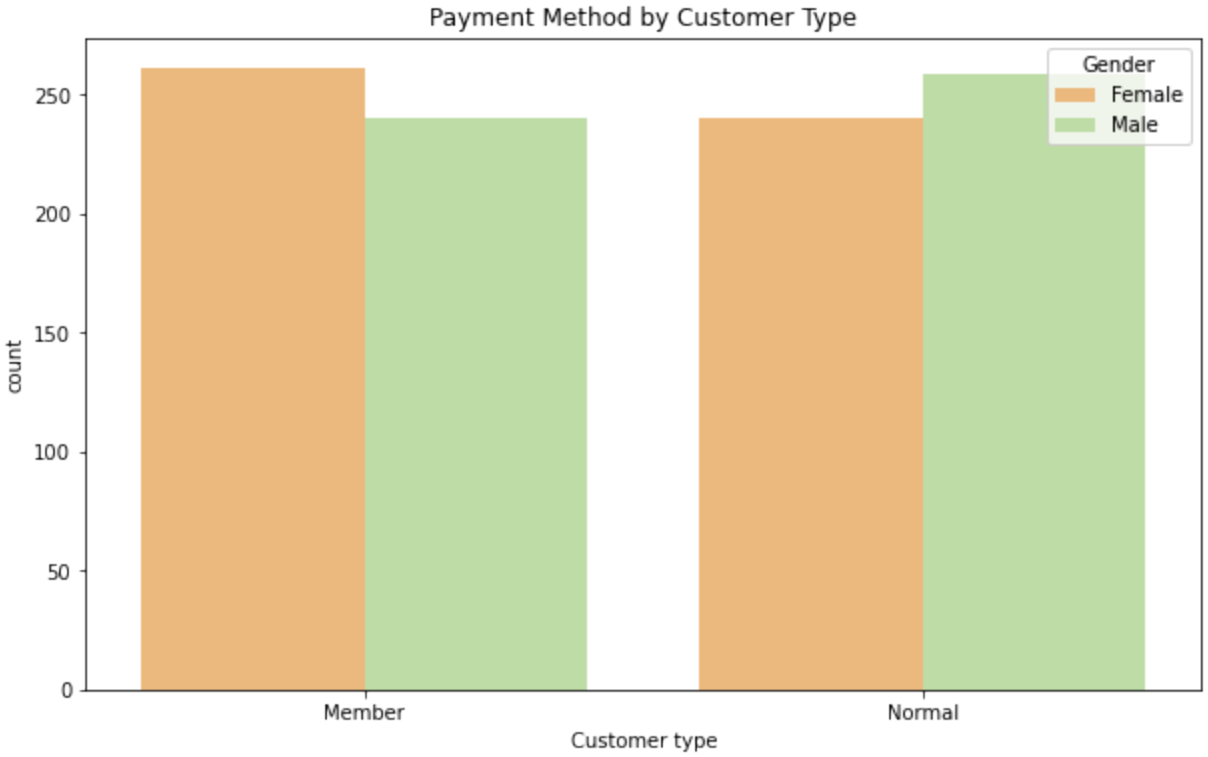


Sales Distribution by Customer type based on Membership

In [57]:
```
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x='Customer type', palette='Set1')
plt.title('Customer Type Distribution')
plt.show()
```

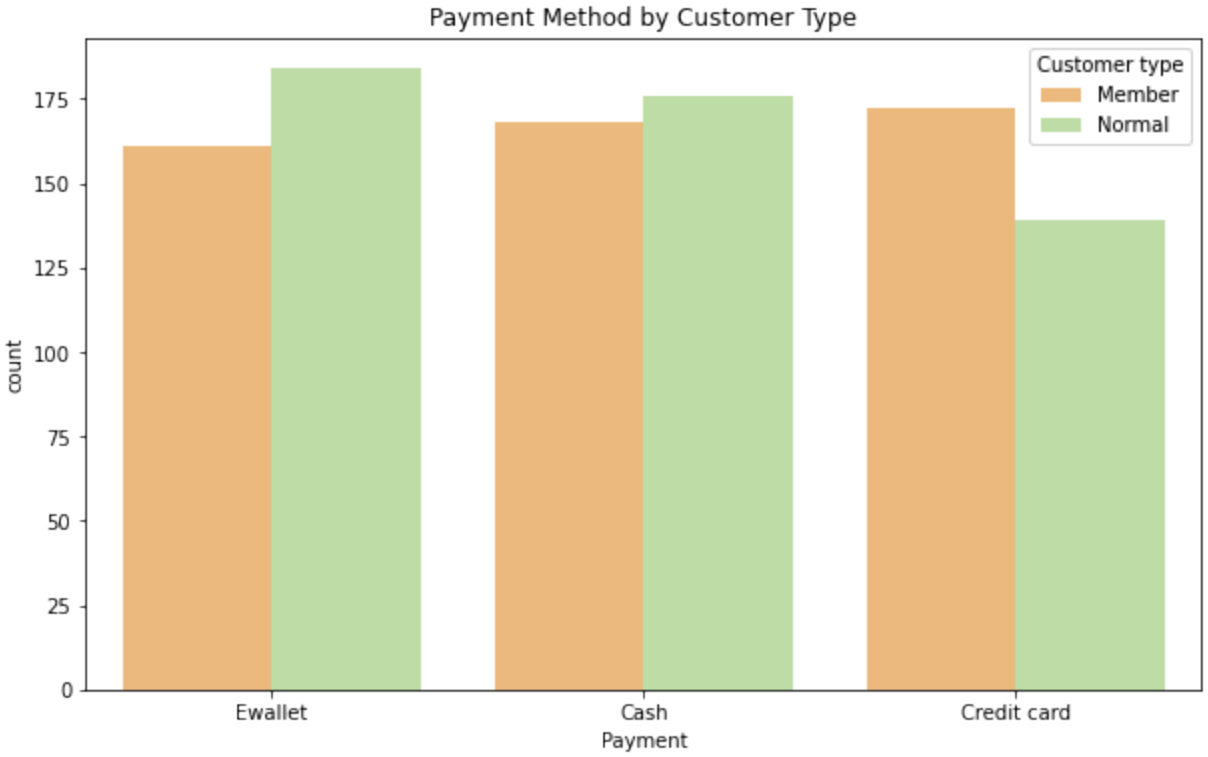
Which Gender tends to invest in a Membership?

In [45]:
```
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x='Customer type', hue='Gender', palette='Spectral')
plt.title('Customer Type by Gender')
plt.show()
```



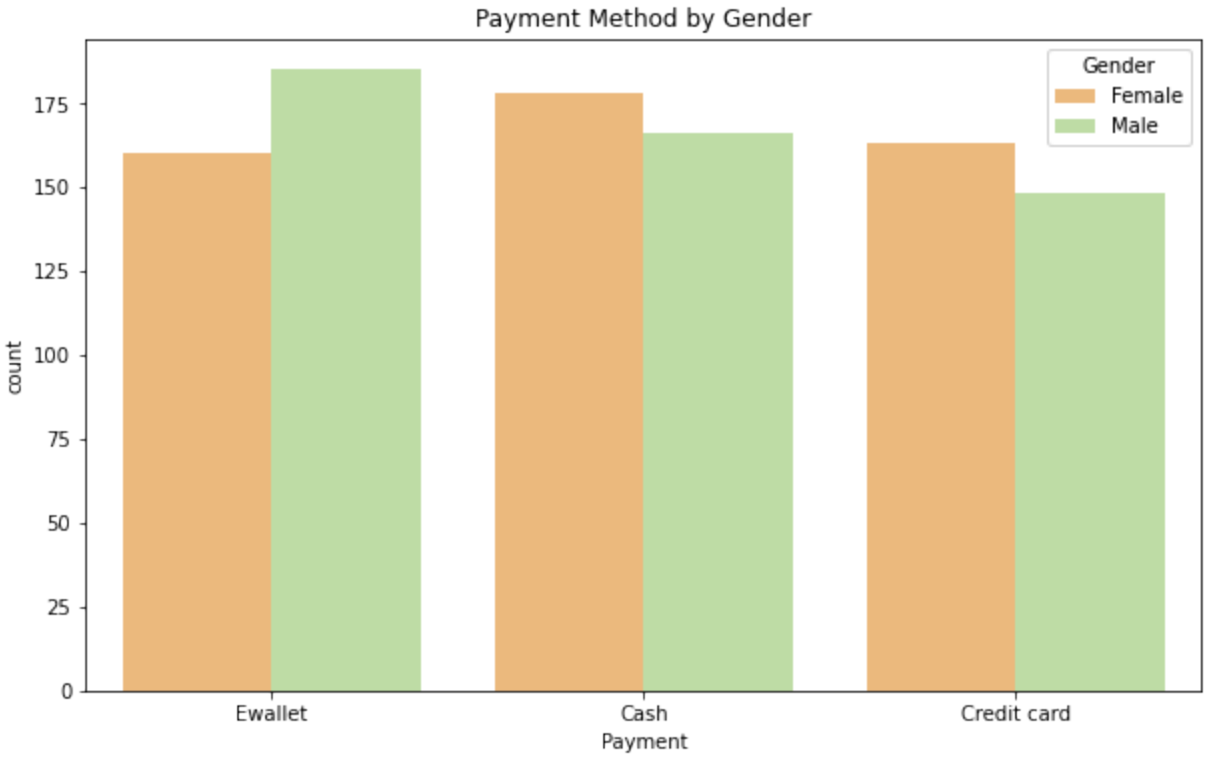
Payment Method by Customer Type

In [43]:
```
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x='Payment', hue='Customer type', palette='Spectral')
plt.title('Payment Method by Customer Type')
plt.show()
```



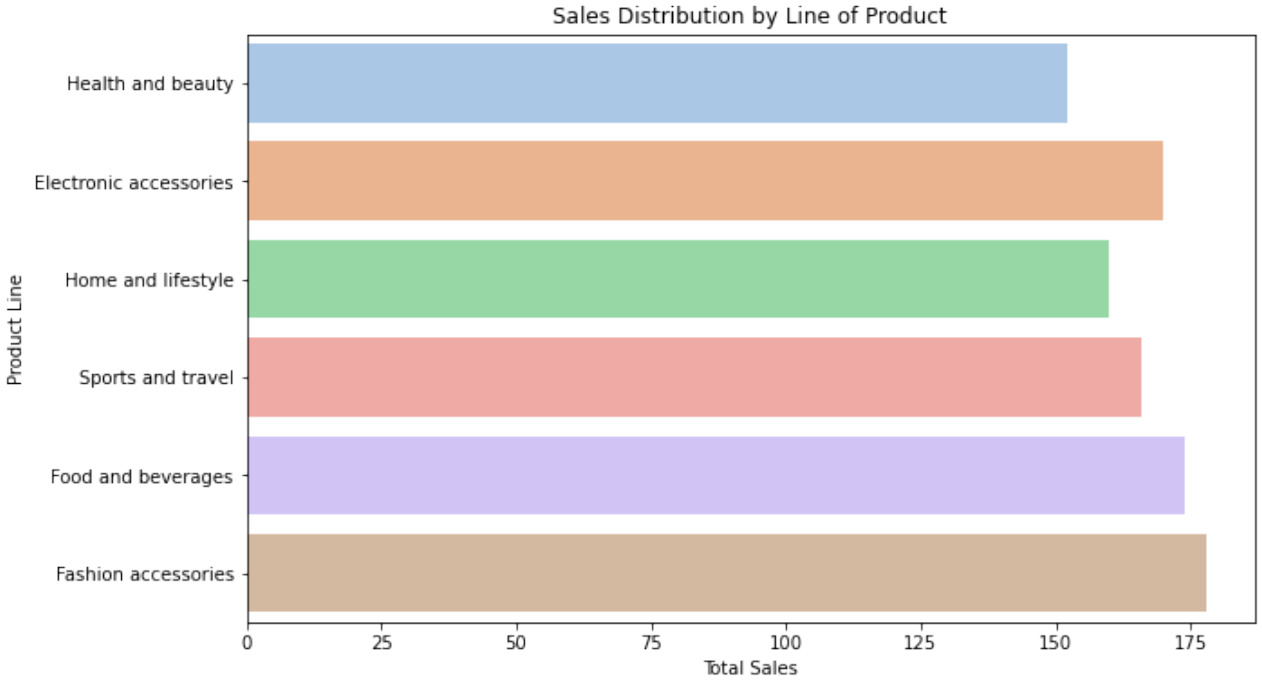
Payment Method by Gender

In [41]:
```
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x='Payment', hue='Gender', palette='Spectral')
plt.title('Payment Method by Gender')
plt.show()
```



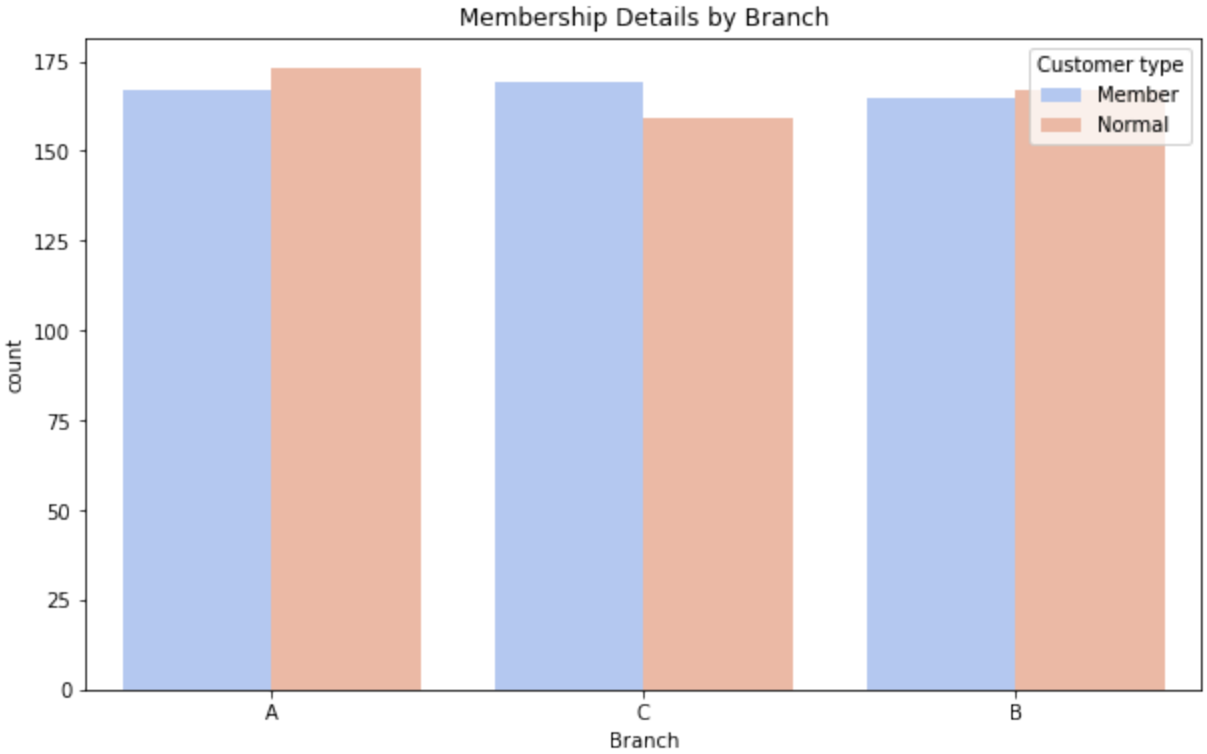
Sales Distribution by Line of Product

In [40]:
```
plt.figure(figsize=(10, 6))
sns.countplot(data=df, y='Product line', palette='pastel')
plt.title('Sales Distribution by Line of Product')
plt.xlabel('Total Sales')
plt.ylabel('Product Line')
plt.show()
```



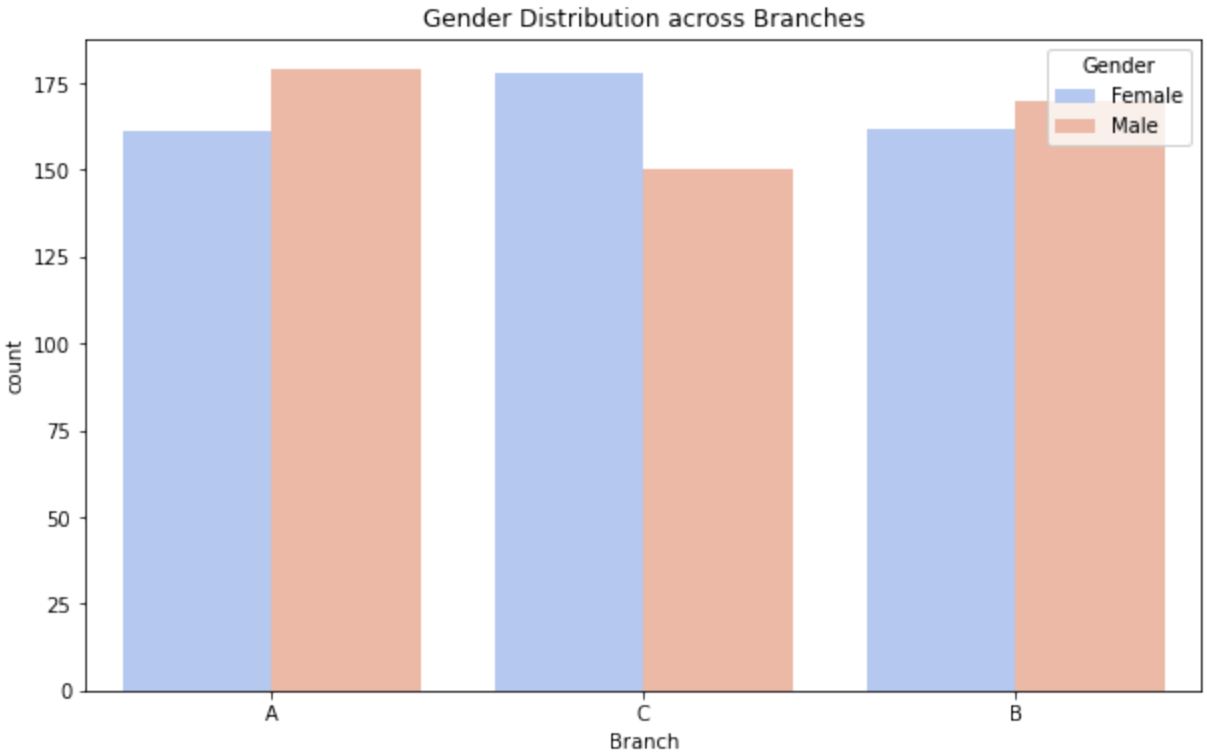
Membership Details by Branch

In [67]:
```
plt.figure(figsize=(10, 6))
sns.countplot(data=df,  x='Branch', hue='Customer type', palette='coolwarm')
plt.title('Membership Details by Branch')
plt.show()
```



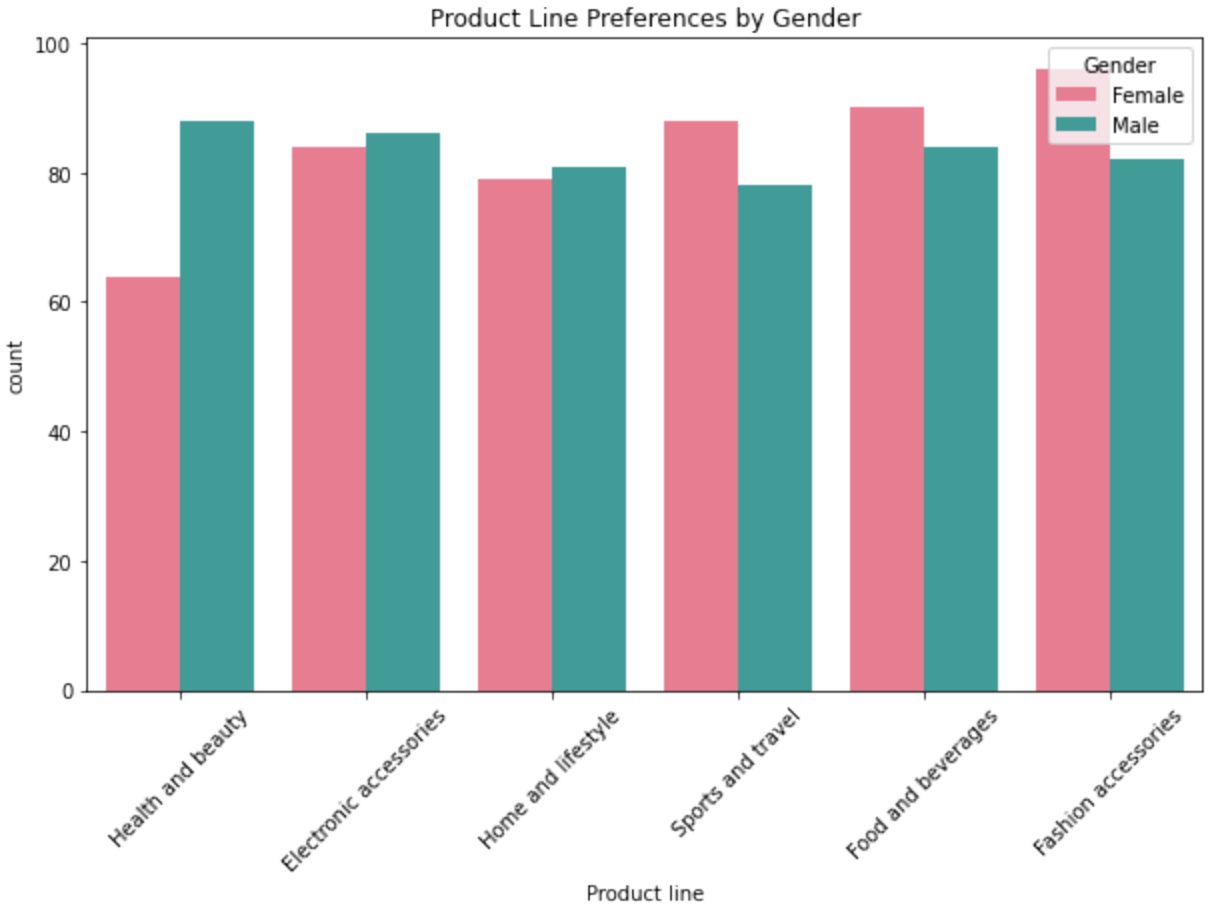
Gender Distribution across Branches

In [66]:
```
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x='Branch', hue='Gender', palette='coolwarm')
plt.title('Gender Distribution across Branches')
plt.show()
```



Product Line Preferences by Gender

In [71]:
```
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x='Product line', hue='Gender', palette='husl')
plt.xticks(rotation=45)
plt.title('Product Line Preferences by Gender')
plt.show()
```

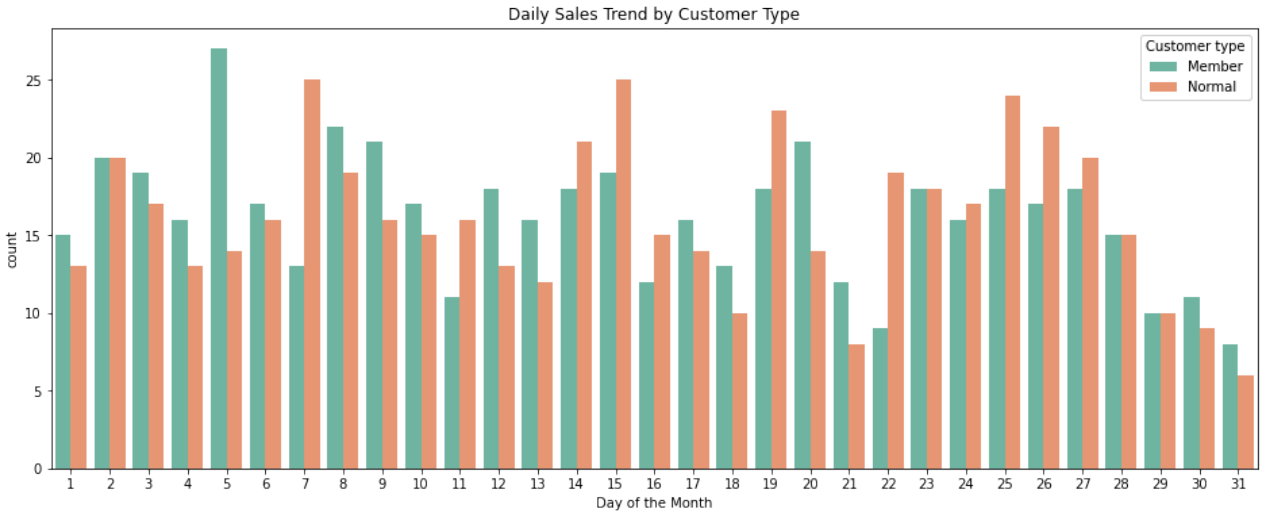


In [72]:
```
import datetime
```

In [73]:
```
# Converting 'Date' and 'Time' to appropriate formats if not done
df['Date'] = pd.to_datetime(df['Date'])
df['Time'] = pd.to_datetime(df['Time']).dt.hour  # Convert to hour for easier visualiza
```

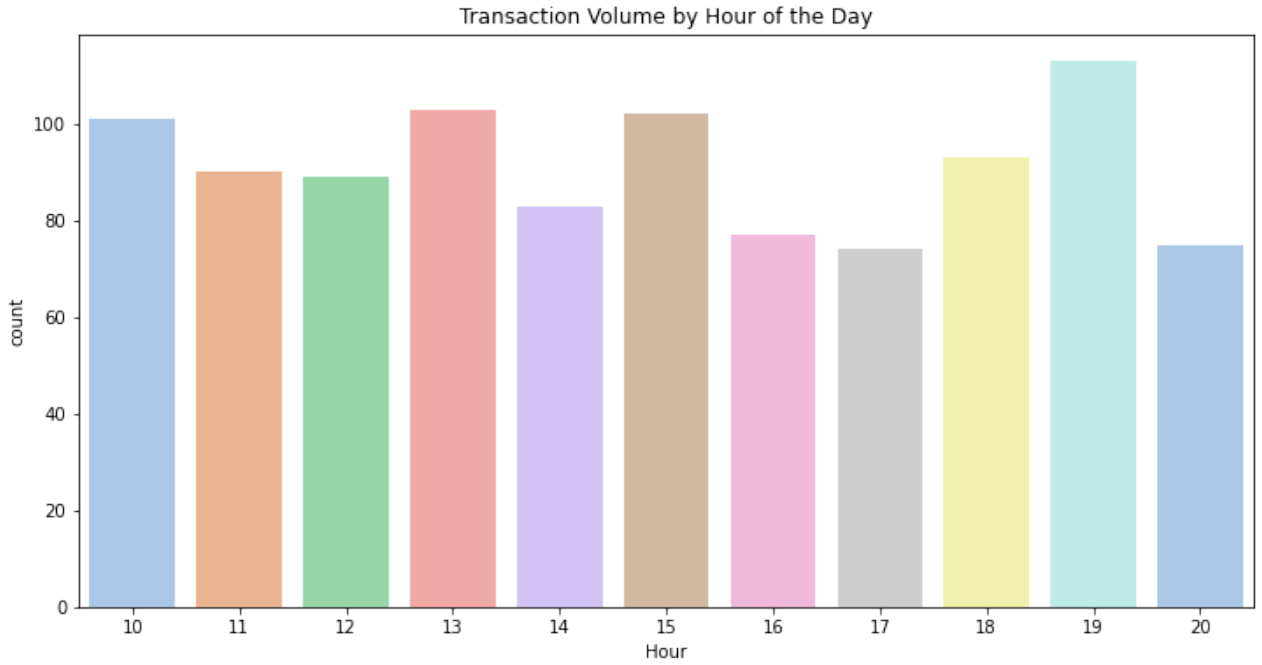
Daily Sales Trend by Customer Type

In [75]:
```
plt.figure(figsize=(16, 6))
df['Day'] = df['Date'].dt.day
sns.countplot(data=df, x='Day', hue='Customer type', palette='Set2')
plt.title('Daily Sales Trend by Customer Type')
plt.xlabel("Day of the Month")
plt.show()
```



Transaction Volume by Hour of the Day

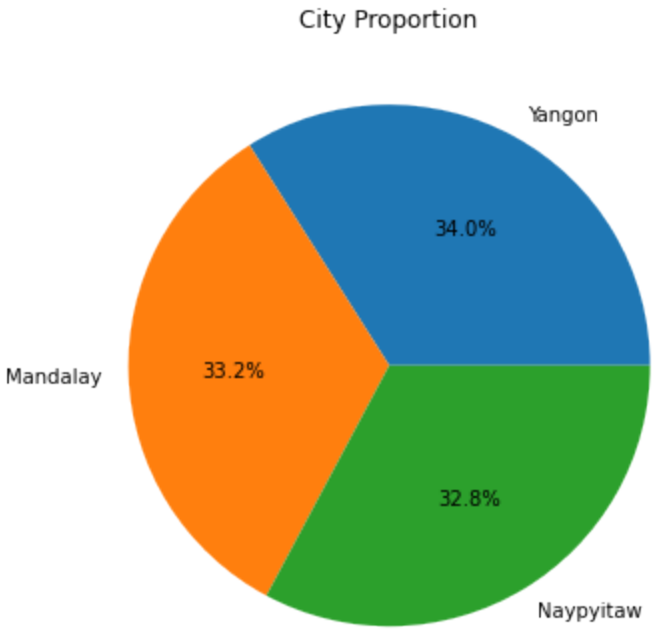
In [77]:
```
plt.figure(figsize=(12, 6))
sns.countplot(data=df, x='Time', palette='pastel')
plt.title('Transaction Volume by Hour of the Day')
plt.xlabel("Hour")
plt.show()
```



In [78]:
```
print(f"Number of unique Invoice IDs: {df['Invoice ID'].nunique()}")
```

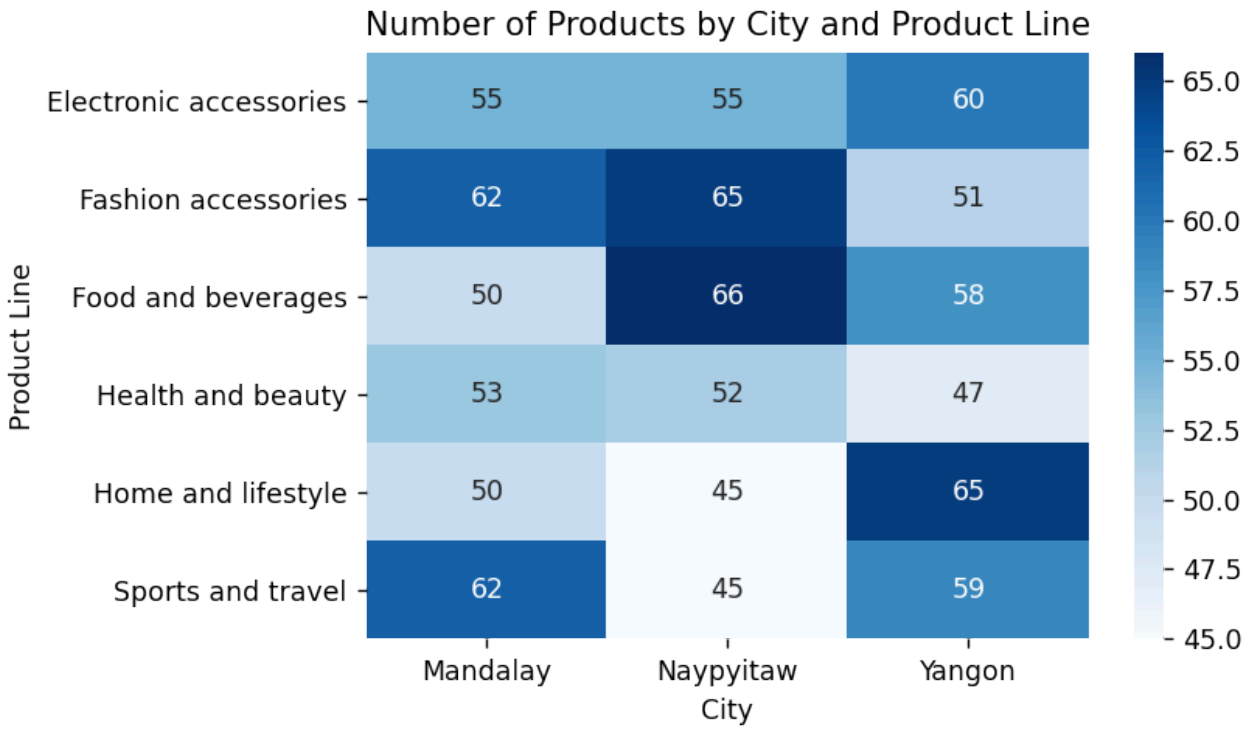
```
Number of unique Invoice IDs: 1000
```

In [79]:
```
plt.figure(figsize=(6, 6))
df['City'].value_counts().plot.pie(autopct='%1.1f%%')
plt.title('City Proportion')
plt.ylabel('')
plt.show()
```



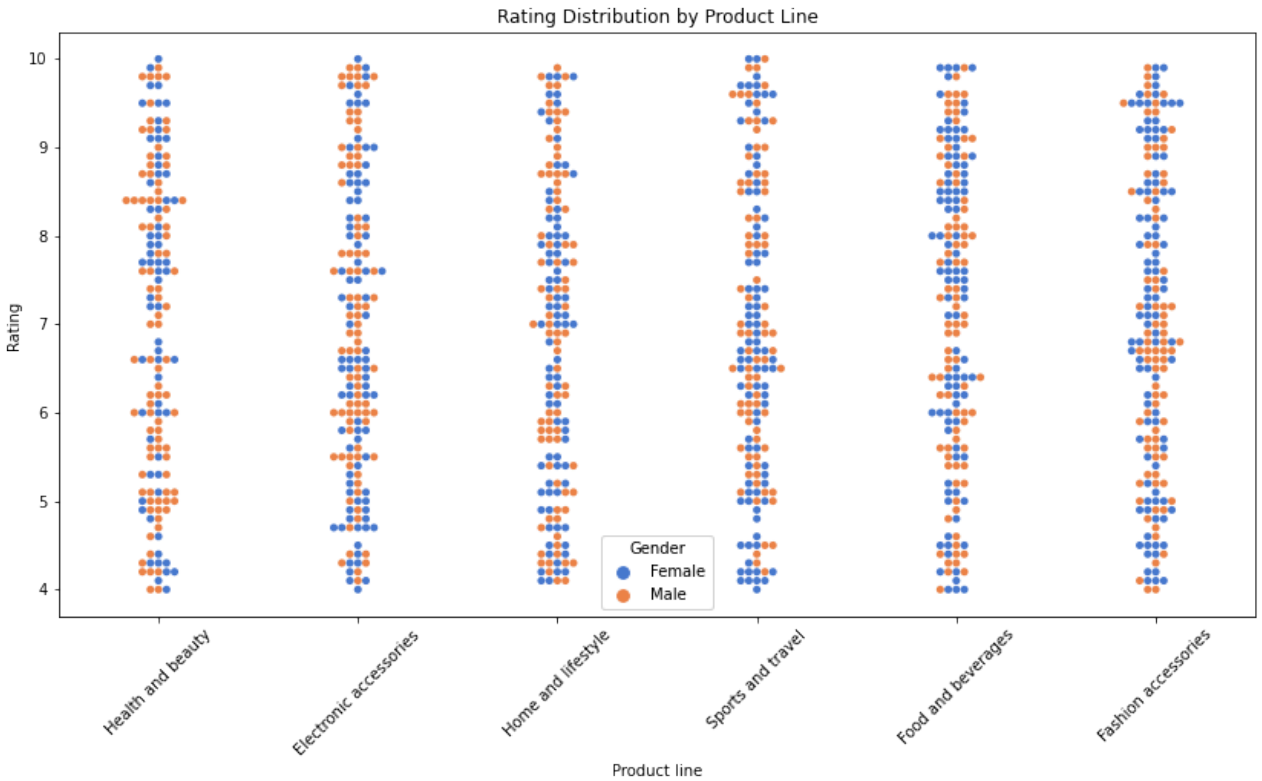
Number of Products by City and Product Line

In [95]:
```
# Aggregate the data: count the occurrences of each City per Product line
heatmap_data = pd.crosstab(index=df['Product line'], columns=df['City'])

# Create the heatmap
plt.figure(dpi=125)
sns.heatmap(heatmap_data, annot=True, cmap="Blues", fmt="d")

# Set labels and title
plt.title('Number of Products by City and Product Line')
plt.xlabel('City')
plt.ylabel('Product Line')

# Show the plot
plt.show()
```



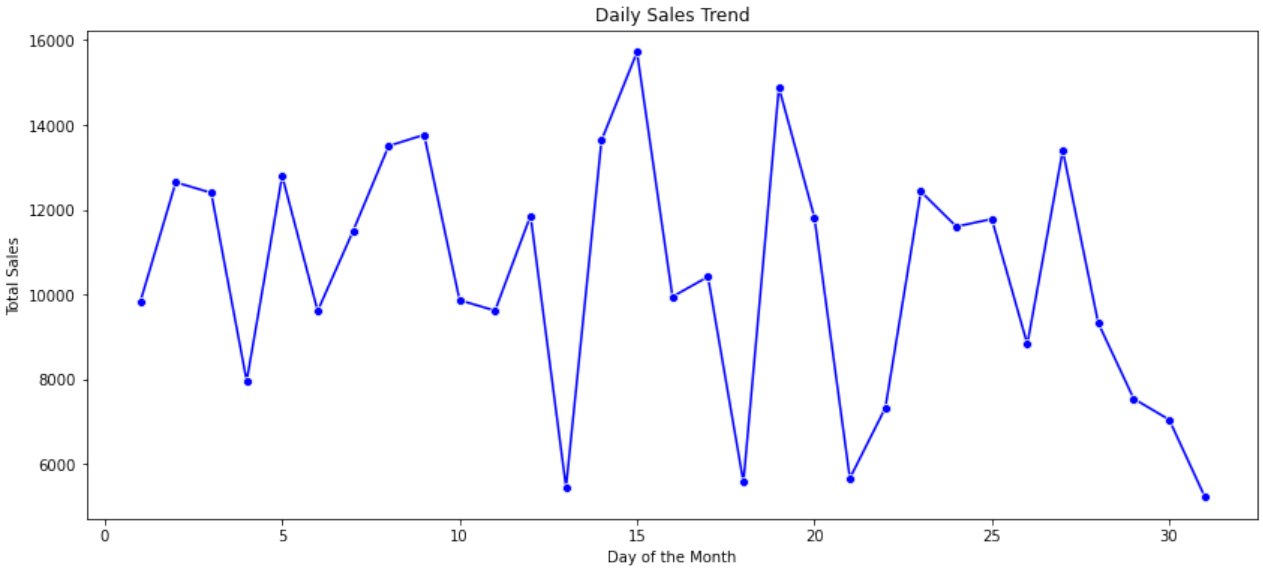
Rating Distribution by Product Line

In [81]:
```
plt.figure(figsize=(14, 7))
sns.swarmplot(data=df, x='Product line', y='Rating', hue='Gender', palette='muted')
plt.xticks(rotation=45)
plt.title('Rating Distribution by Product Line')
plt.show()
```



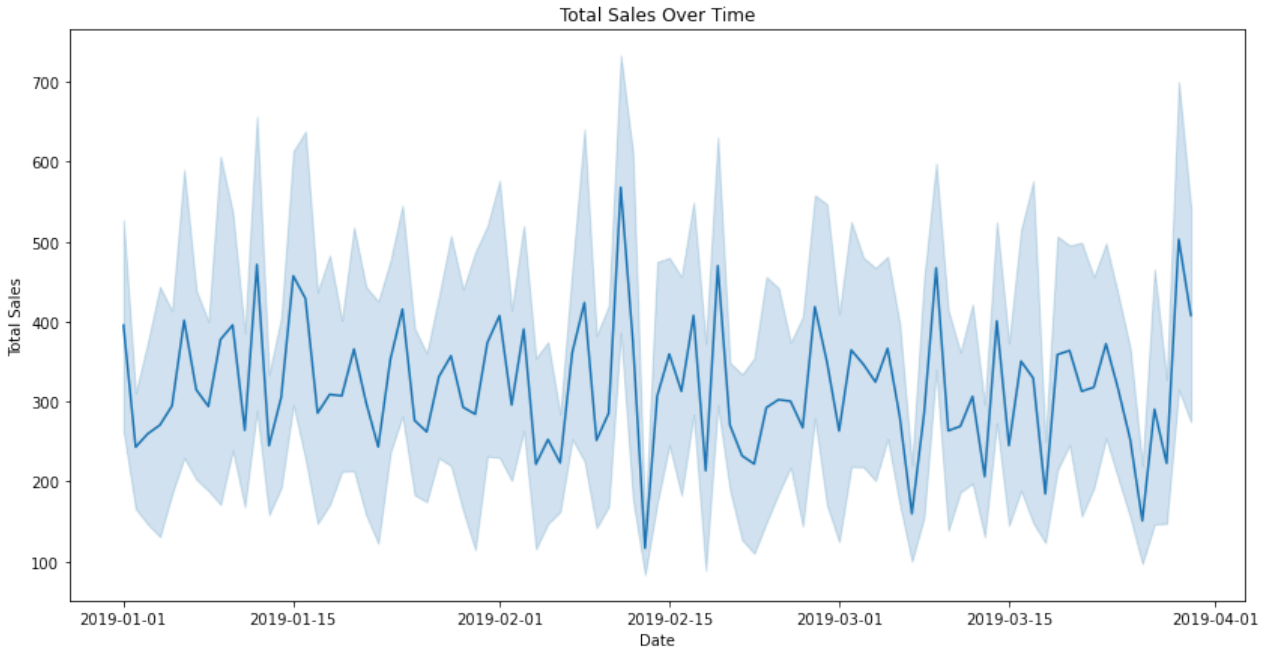
Daily Sales Trend

In [82]:
```
plt.figure(figsize=(14, 6))
df['Day'] = df['Date'].dt.day
daily_sales = df.groupby('Day')['Total'].sum()
sns.lineplot(x=daily_sales.index, y=daily_sales.values, marker='o', color='b')
plt.title('Daily Sales Trend')
plt.xlabel('Day of the Month')
plt.ylabel('Total Sales')
plt.show()
```



Total Sales Over Time

In [84]:
```
df['Date'] = pd.to_datetime(df['Date'])
plt.figure(figsize=(14, 7))
sns.lineplot(x=df['Date'], y=df['Total'])
plt.title("Total Sales Over Time")
plt.xlabel("Date")
plt.ylabel("Total Sales")
plt.show()
```



Product line sold most on that Date

In [93]:
```
plt.figure(figsize=(14, 7))
sns.lineplot(x=df['Date'], y=df['Product line'])
plt.title("Sales Over Time based on Product line")
plt.xlabel("Date")
plt.ylabel("Total Sales")
plt.show()
```