Big Basket Mini Project

E-commerce (electronic commerce) is the activity of electronically buying or selling of products on online services or over the Internet. E- commerce draws on technologies such as mobile commerce, electronic funds transfer, supply chain management, Internet marketing, online transaction processing, electronic data interchange (EDI), inventory management systems, and automated data collection systems. E- commerce is in turn driven by the technological advances of the semiconductor industry, and is the largest sector of the electronics industry.

Big basket is the largest online grocery supermarket in India. Was launched somewhere around in 2011 since then they have been expanding their business. Though some new competitors have been able to set their foot in the nation such as Blink it etc. but Big Basket has still not lost anything.

This dataset contains 10 attributes with simple meaning and which are described as follows:

- index - Simply the Index!
- product - Title of the product (as they are listed)
- category - Category into which product has been classified
- sub _ category - Subcategory into which product has been kept
- brand - Brand of the product
- sale _price - Price at which product is being sold on the site
- market _price - Market price of the product
- type - Type into which product falls
- rating - Rating the product has got from its consumers
- description - Description of the dataset (in detail)

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

Step 1: Load DataSet

In [ ]:
```
data = pd.read_csv("BigBasket Products.csv")
```

Step 2: Use head function to look for first 12 rows

In [ ]:
```
data.head(12)
```

Out[ ]:

	index	product	category	sub_category	brand	sale_price	m
0	1	Garlic Oil - Vegetarian Capsule 500 mg	Beauty & Hygiene	Hair Care	Sri Sri Ayurveda	220.0	
1	2	Water Bottle - Orange	Kitchen, Garden & Pets	Storage & Accessories	Mastercook	180.0	
2	3	Brass Angle Deep - Plain, No.2	Cleaning & Household	Pooja Needs	Trm	119.0	
3	4	Cereal Flip Lid Container/Storage Jar - Assort...	Cleaning & Household	Bins & Bathroom Ware	Nakoda	149.0	
4	5	Creme Soft Soap - For Hands & Body	Beauty & Hygiene	Bath & Hand Wash	Nivea	162.0	
5	6	Germ - Removal Multipurpose Wipes	Cleaning & Household	All Purpose Cleaners	Nature Protect	169.0	
6	7	Multani Mati	Beauty & Hygiene	Skin Care	Satinance	58.0	
7	8	Hand Sanitizer - 70% Alcohol Base	Beauty & Hygiene	Bath & Hand Wash	Bionova	250.0	
8	9	Biotin & Collagen Volumizing Hair Shampoo + Bi...	Beauty & Hygiene	Hair Care	StBotanica	1098.0	
9	10	Scrub Pad - Anti-Bacterial, Regular	Cleaning & Household	Mops, Brushes & Scrubs	Scotch brite	20.0	
10	11	Wheat Grass Powder - Raw	Gourmet & World Food	Cooking & Baking Needs	NUTRASHIL	261.0	
11	12	Butter Cookies Gold Collection	Gourmet & World Food	Chocolates & Biscuits	Sapphire	600.0	

Step 3: Get Description of the data in the DataFrame

In [ ]:
```
data.describe()
```

Out[ ]:

	index	sale_price	market_price	rating
count	27555.00000	27549.000000	27555.000000	18919.000000
mean	13778.00000	334.648391	382.056664	3.943295
std	7954.58767	1202.102113	581.730717	0.739217
min	1.00000	2.450000	3.000000	1.000000
25%	6889.50000	95.000000	100.000000	3.700000
50%	13778.00000	190.320000	220.000000	4.100000
75%	20666.50000	359.000000	425.000000	4.300000
max	27555.00000	112475.000000	12500.000000	5.000000

Step 4: Find Information about the DataFrame

In [ ]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 27555 entries, 0 to 27554
Data columns (total 10 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   index         27555 non-null  int64  
 1   product       27554 non-null  object 
 2   category      27555 non-null  object 
 3   sub_category  27555 non-null  object 
 4   brand         27554 non-null  object 
 5   sale_price    27549 non-null  float64
 6   market_price  27555 non-null  float64
 7   type          27555 non-null  object 
 8   rating        18919 non-null  float64
 9   description   27440 non-null  object 
dtypes: float64(3), int64(1), object(6)
memory usage: 2.1+ MB
```

Step 5: Find out Top & least sold products

In [ ]:
```
## assuming as per sale price highest and lowest

top = data.sort_values(by='sale_price', ascending=False).head(5)
least = data.sort_values(by='sale_price', ascending=True).head(5)
print("Top 5 sold Products (As proxy for Top sold):")
top[['product', 'sale_price']]
```

```
Top 5 sold Products (As proxy for Top sold):
```

Out[ ]:

	product	sale_price
1249	Beard Kit	112475.0
248	4mm Aluminium Induction Base Chapati Roti Tawa...	111649.0
436	Balloon - Polka Dot, 12 Inch	88899.0
288	Arrabbiata Tomato Pasta Sauce With Chilli	22325.0
25301	Bravura Clipper	12500.0

In [ ]:
```
print("Least 5 Sold Products (As proxy for Least sold):")
least[['product', 'sale_price']]
```

```
Least 5 Sold Products (As proxy for Least sold):
```

Out[ ]:

	product	sale_price
26976	Curry Leaves	2.45
21312	Serum	3.00
2978	Sugar Free Chewing Gum - Mixed Fruit	5.00
27490	50-50 Timepass Salted Biscuits	5.00
22655	Chewing Gum - Peppermint	5.00

Step 6: Measuring discount on a certain item

In [ ]:
```
data['discount'] = data['market_price'] - data['sale_price']
data[['product', 'market_price', 'sale_price', 'discount']]
```

Out[ ]:

	product	market_price	sale_price	discount
0	Garlic Oil - Vegetarian Capsule 500 mg	220.0	220.00	0.00
1	Water Bottle - Orange	180.0	180.00	0.00
2	Brass Angle Deep - Plain, No.2	250.0	119.00	131.00
3	Cereal Flip Lid Container/Storage Jar - Assort...	176.0	149.00	27.00
4	Creme Soft Soap - For Hands & Body	162.0	162.00	0.00
...	...	...	...	...
27550	Wottagirl! Perfume Spray - Heaven, Classic	249.0	199.20	49.80
27551	Rosemary	75.0	67.50	7.50
27552	Peri-Peri Sweet Potato Chips	200.0	200.00	0.00
27553	Green Tea - Pure Original	495.0	396.00	99.00
27554	United Dreams Go Far Deodorant	390.0	214.53	175.47

27555 rows × 4 columns

Step 7: Find out the Missing Values from the Dataset

In [ ]:
```
data.isnull().sum()
```

Out[ ]:

	0
index	0
product	1
category	0
sub_category	0
brand	1
sale_price	6
market_price	0
type	0
rating	8636
description	115
discount	6

dtype: int64

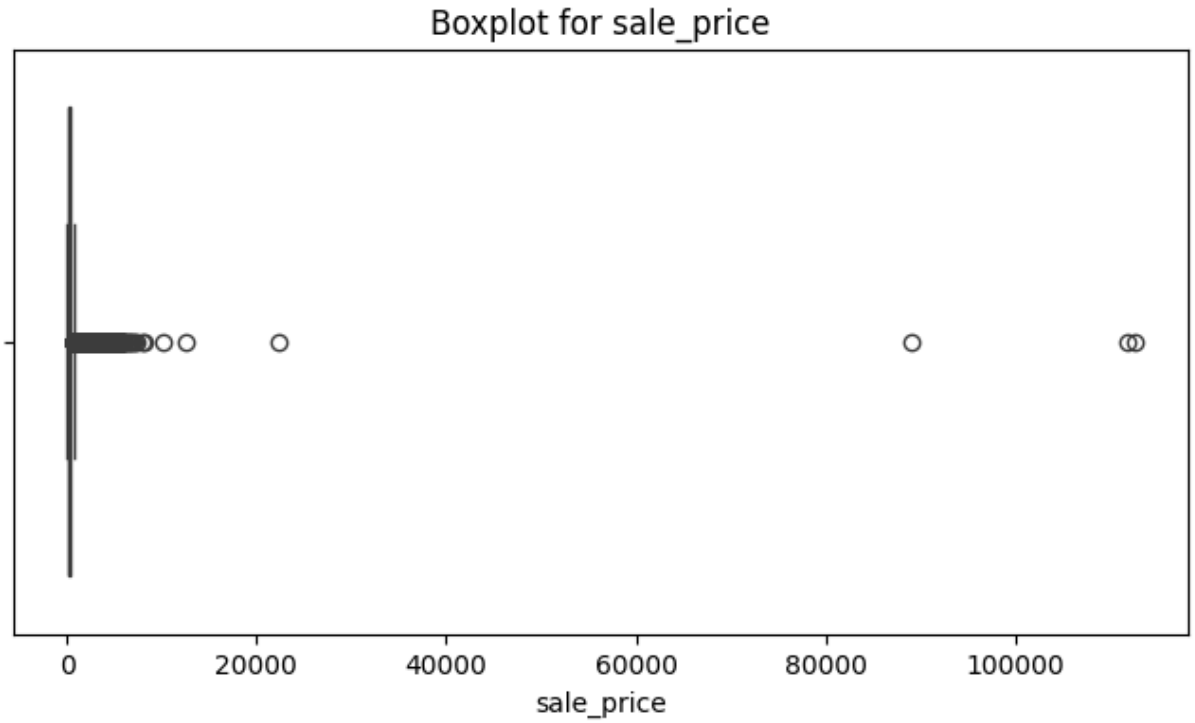
Step 8: Find out the outliers from the dataset according to

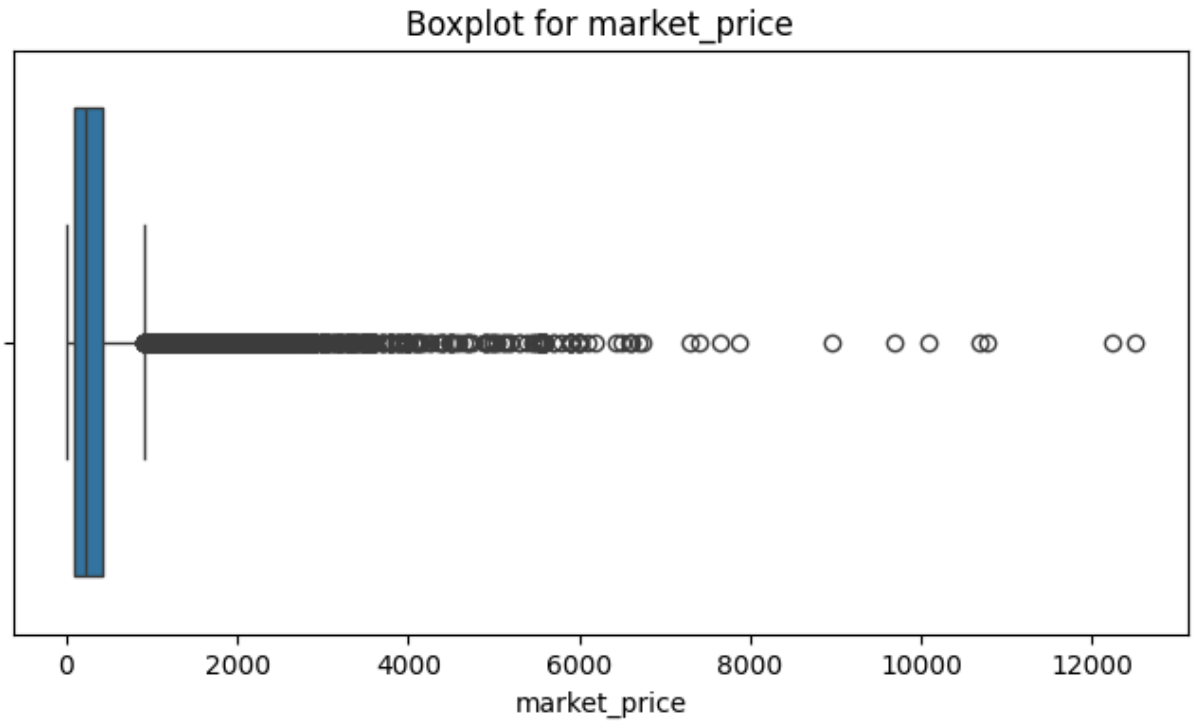
the columns and fill them with the mean

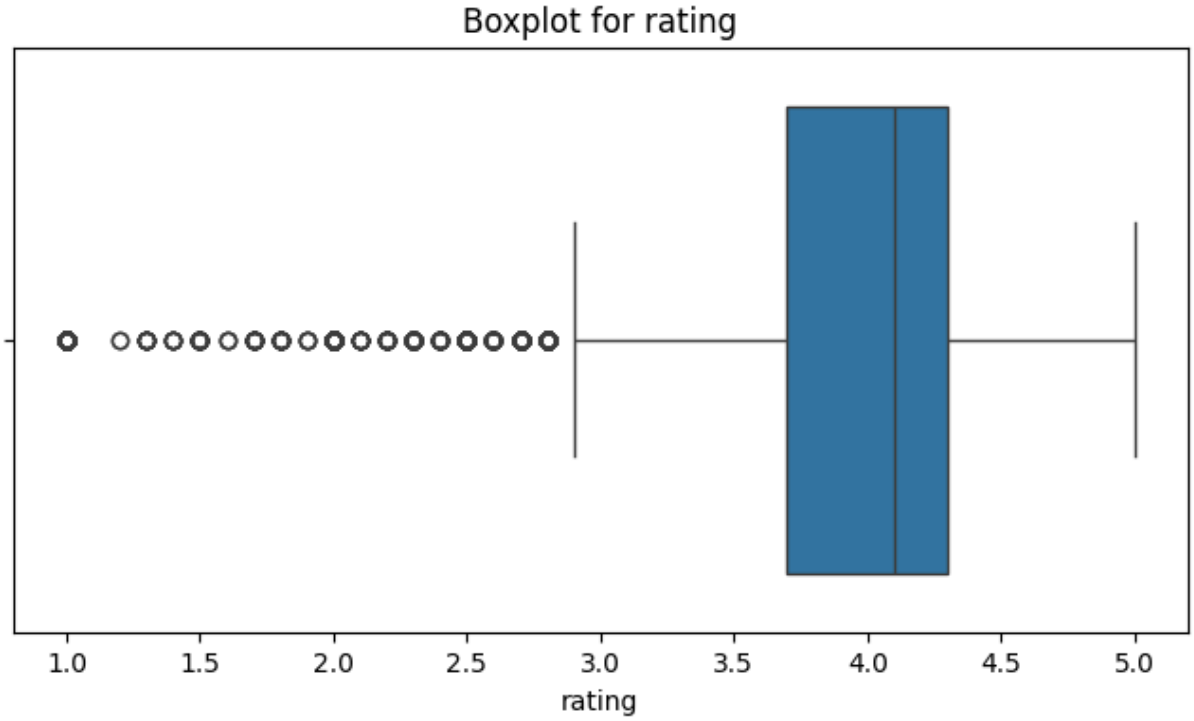
1. FINDING OUTLIERS

In [ ]:
```
cols = ['sale_price', 'market_price', 'rating']

for col in cols:
    plt.figure(figsize=(8, 4))
    sns.boxplot(x=data[col])
    plt.title(f'Boxplot for {col}')
    plt.show()
```







Insight:

- Sale Price & Market Price: Both show a right-skewed distribution with significant outliers on the higher end. This suggests a small portion of products are priced very high, possibly premium or bulk items.
- Rating: Ratings are mostly concentrated around 3.5 to 5. Some products have very low ratings, indicating either quality concerns or low user engagement.

2. FIXING OUTLIERS

In [ ]:
```
for col in cols:
    q1 = data[col].quantile(0.25)
    q3 = data[col].quantile(0.75)
    iqr = q3 - q1
    lower_bound = q1 - 1.5 * iqr
    upper_bound = q3 + 1.5 * iqr
    mean = data[col].mean()

    # Replace outliers with mean
    data[col] = np.where((data[col] < lower_bound) | (data[col] > upper_bour
```

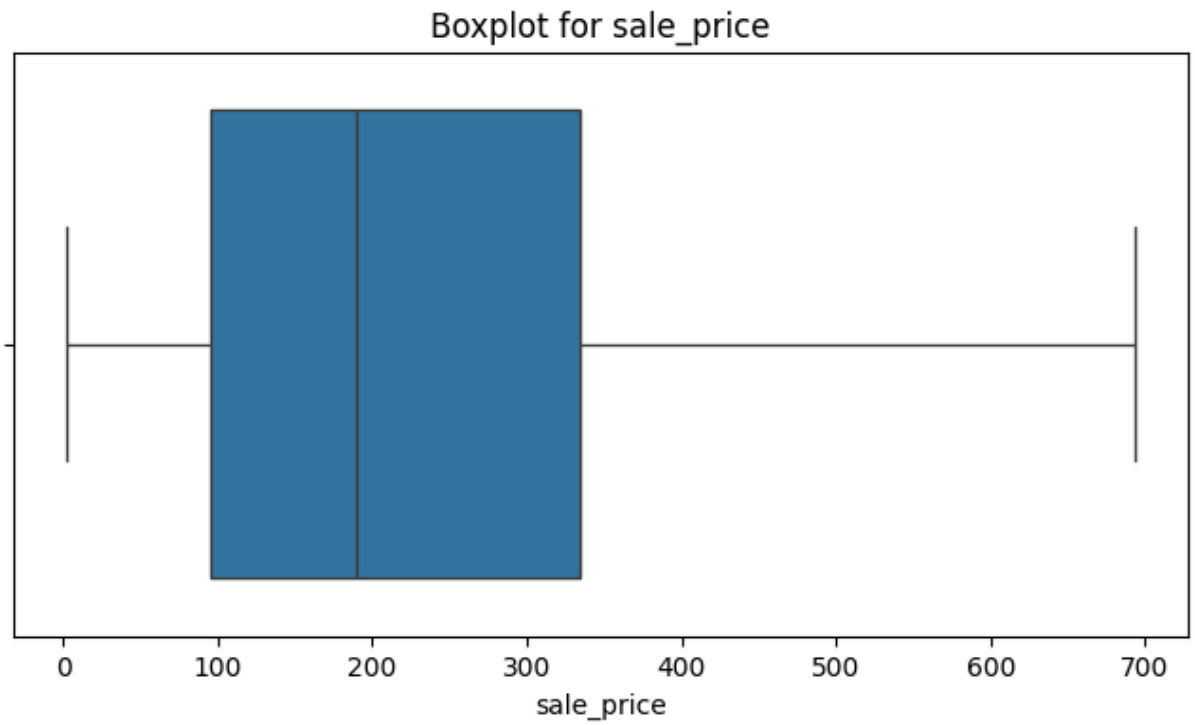
In [ ]:
```
print("Outliers identified via boxplots and replaced with column mean.")
```

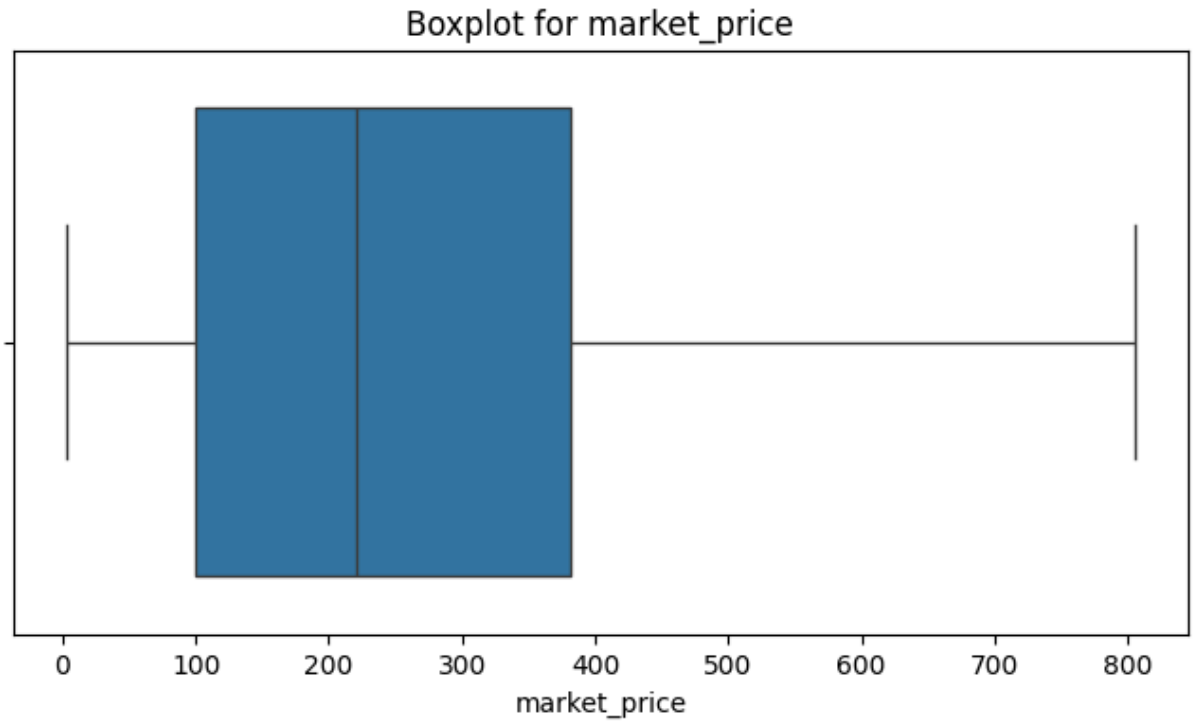
```
Outliers identified via boxplots and replaced with column mean.
```

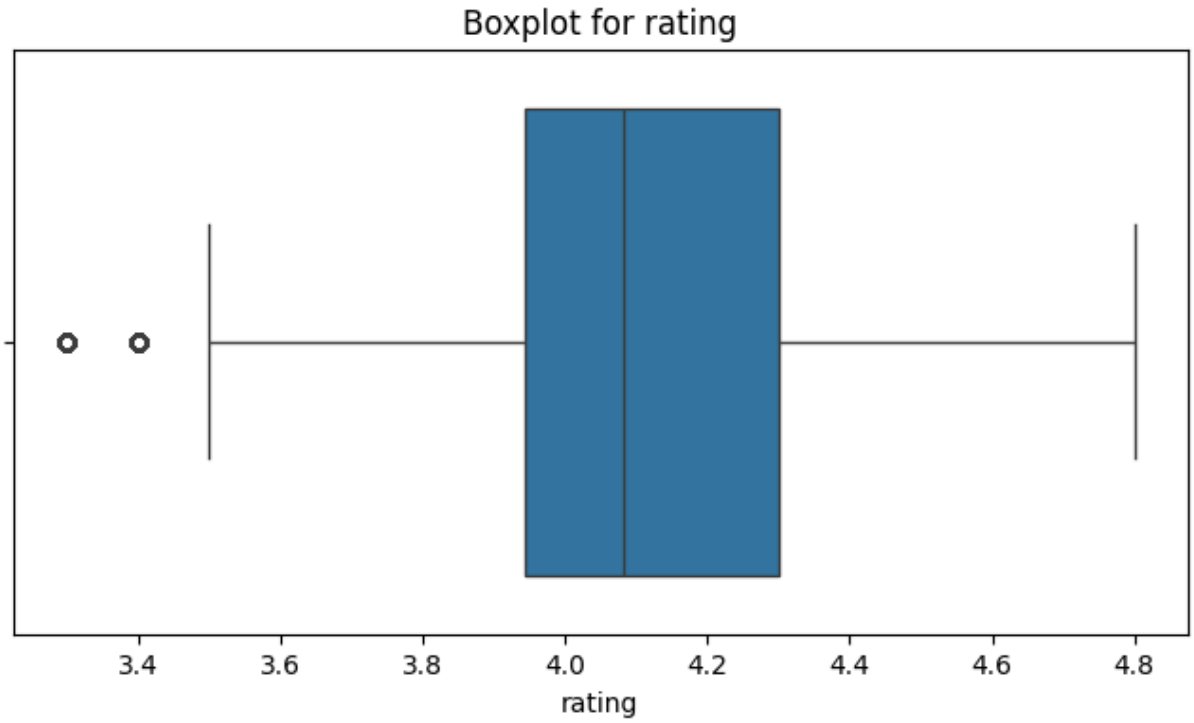
CHECKING FOR OUTLIERS AGAIN

In [ ]:
```
cols = ['sale_price', 'market_price', 'rating']

for col in cols:
    plt.figure(figsize=(8, 4))
    sns.boxplot(x=data[col])
    plt.title(f'Boxplot for {col}')
```



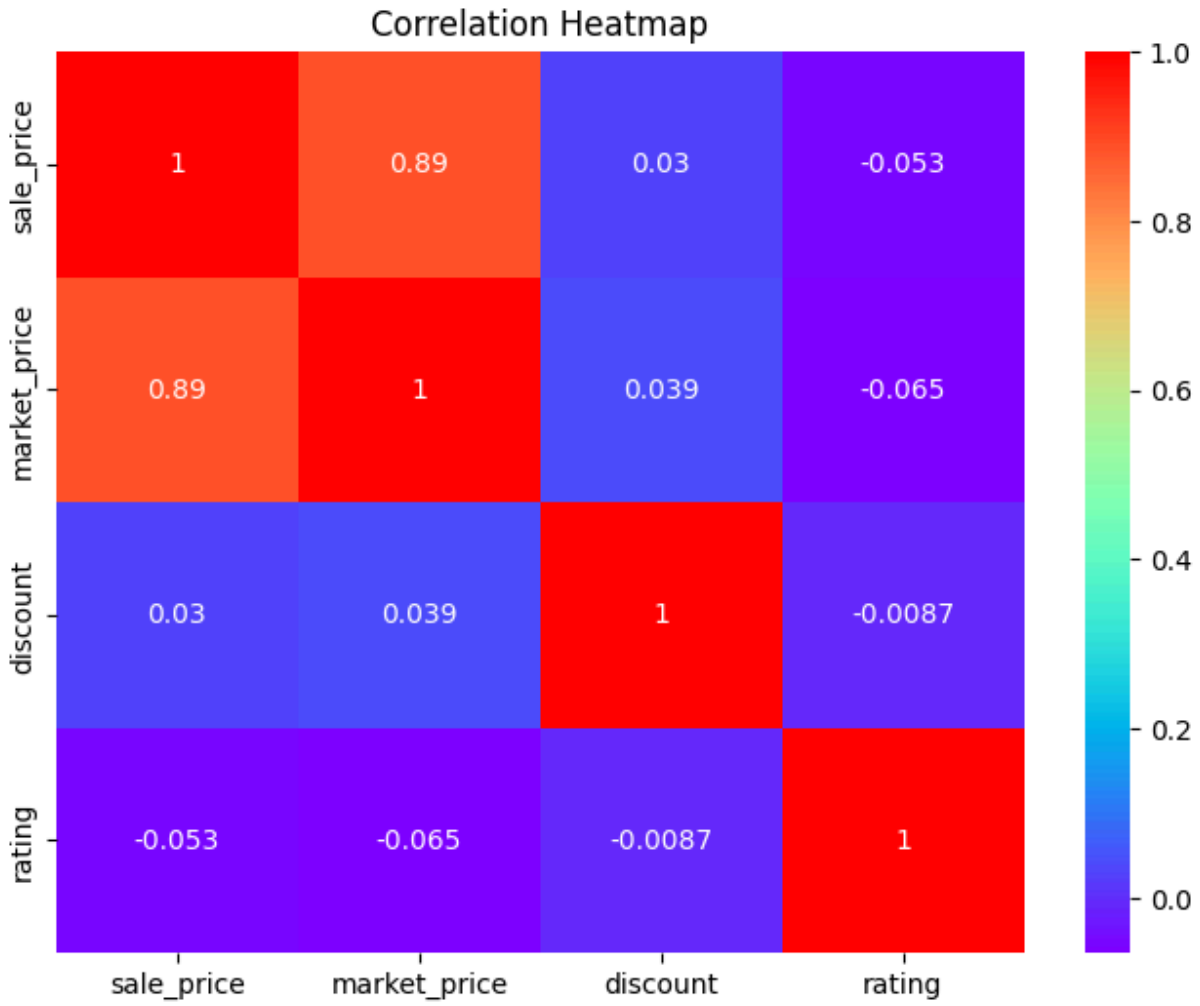




Step 9: Create Plots or visualizations

In [ ]:
```
plt.figure(figsize=(8, 6))
sns.heatmap(data[['sale_price', 'market_price', 'discount', 'rating']].corr(
plt.title('Correlation Heatmap')
```

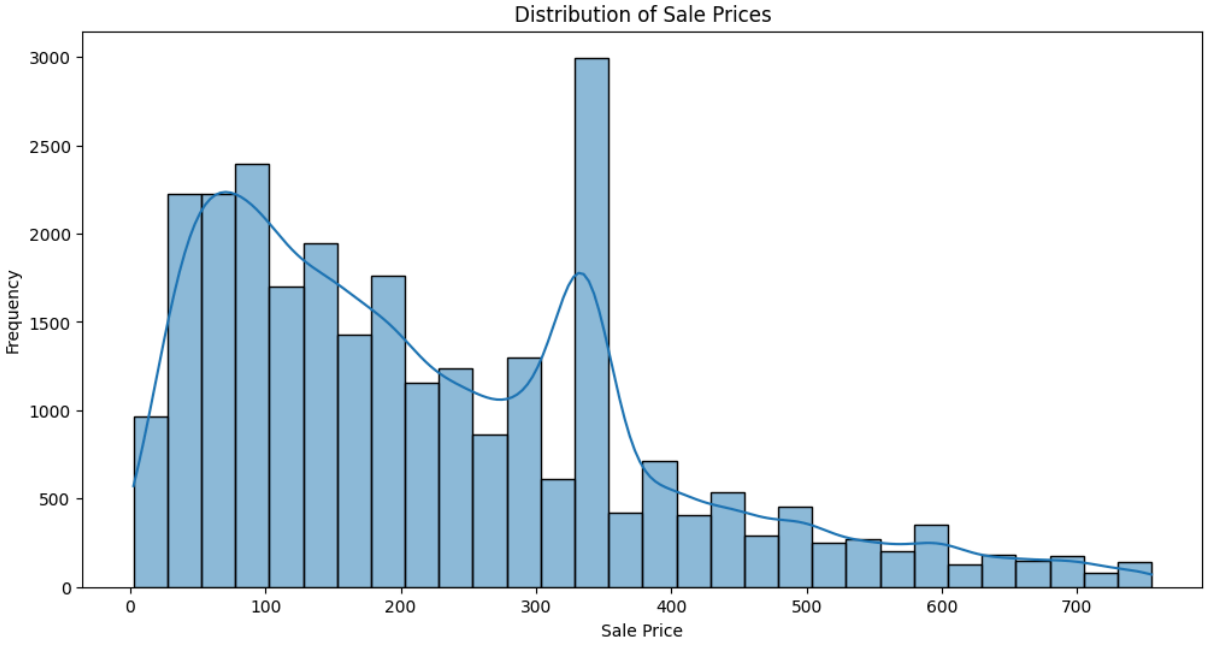
Out[ ]:
```
Text(0.5, 1.0, 'Correlation Heatmap')
```



Insight:

- High correlation between sale_price and market_price: This is expected since discounts are applied on MRP.
- Discount vs Price: Discounts are negatively correlated with sale/market price, indicating costlier items tend to have relatively lower discounts.
- Rating has weak correlations with price and discount, showing price doesn't strongly influence customer ratings.

In [ ]:
```
plt.figure(figsize=(12, 6))
sns.histplot(data['sale_price'], bins=30, kde=True)
plt.title('Distribution of Sale Prices')
plt.xlabel('Sale Price')
plt.ylabel('Frequency')
plt.show()
```

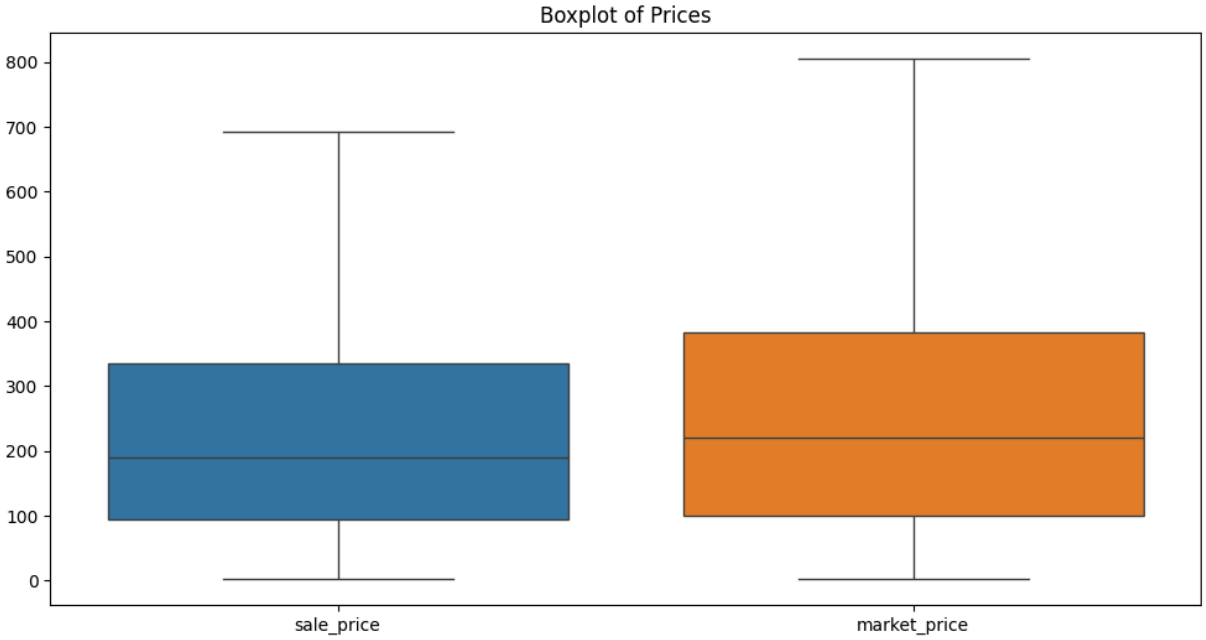


Insight:

- The distribution is right-skewed, showing most products are priced under a certain threshold (likely ₹100–₹500).
- There are fewer high-priced products, again indicating niche or premium items.

In [ ]:
```
plt.figure(figsize=(12, 6))
sns.boxplot(data=data[['sale_price', 'market_price']])
plt.title('Boxplot of Prices')
```

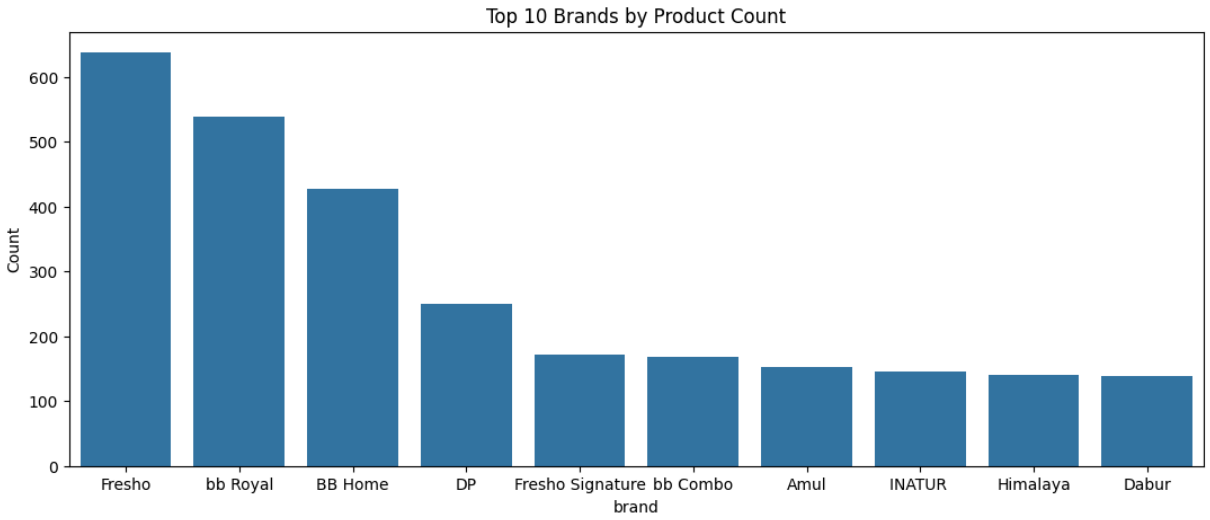
Out[ ]:
```
Text(0.5, 1.0, 'Boxplot of Prices')
```



Insight:

- Confirms that market prices are consistently higher than sale prices. The gap between sale and market price across many products indicates effective discounting strategies by Big Basket.

In [ ]:
```
plt.figure(figsize=(13, 5))
top_brands = data['brand'].value_counts().head(10)
sns.barplot(x=top_brands.index, y=top_brands.values)
plt.title("Top 10 Brands by Product Count")
plt.ylabel("Count")
plt.show()
```



Insight:

- The graph highlights the 10 most frequent brands on Big Basket, based on the number of listed products.
- These brands likely have strong partnerships with Big Basket or broad product ranges (e.g., staples, snacks, beverages).
- A high product count per brand indicates better shelf presence, which may influence customer trust and convenience.
- This distribution can also inform inventory or marketing focus, as these brands contribute heavily to the overall product catalog.

✅ Conclusion – Big Basket Mini Project

The data analysis performed on the Big Basket product listings has provided several key insights into pricing, customer ratings, brand dominance, and product diversity. Here's a consolidated conclusion:

📌 1. Pricing Patterns

- **Sale prices are consistently lower** than market prices, confirming the presence of discounts.
- **Significant outliers** in both `sale_price` and `market_price` suggest the inclusion of high-end or bulk products.
- Most products fall within an **affordable price range**, supporting Big Basket's mass-market focus.

📌 2. Discount & Rating Insights

- **Weak correlation between discounts and ratings** indicates that customer satisfaction is not solely price-driven.
- Discounts are **inversely related to market price**, hinting that cheaper products are more aggressively discounted.

📌 3. Product Ratings

- Ratings tend to cluster around **4 stars**, indicating generally positive customer feedback.
- Presence of low-rated products also reveals scope for **quality control or better customer feedback management**.

📌 4. Brand Dominance

- A small number of brands dominate the catalog, pointing to **strong supplier relationships** or high brand demand.
- These top brands likely drive **significant traffic and sales** for Big Basket.

📌 5. Product Variety

- The broad distribution of prices and brands shows that Big Basket caters to a **wide range of customer segments**, from budget-conscious buyers to premium shoppers.

🎯 Final Remark

This analysis reinforces that Big Basket leverages **strategic discounting, brand variety, and customer satisfaction** to remain competitive in the online grocery market. The insights derived can help improve marketing strategies, product curation, and pricing models for better business outcomes.