Supermarket Analysis

By:
Robert Gorman

Abstract:

This supermarket dataset is a collection of data that provides information about transactions that took place in a supermarket. This dataset typically includes information such as the date and time of the transaction, the products that were purchased, the price of each product, the total amount spent on the transaction, and other relevant details.

We will use this dataset to perform data analysis and gain insights into the behavior of the supermarket customers. For example, we could use this supermarket dataset to study the relationship between the day of the week and the total amount spent on transactions, or to analyze the factors that influence customer spending patterns.

Overall, this supermarket dataset can be a valuable tool for understanding and predicting the behavior of supermarket customers, and for making data-driven decisions that can improve the performance of a supermarket business.

Introduction

The supermarket dataset analyzed in this Kaggle notebook contains sales data from a supermarket chain. The goal of this analysis is to perform exploratory data analysis (EDA) on the dataset and build predictive models to gain insights that can help the supermarket business improve its operations and profitability.

Business Overview

The supermarket chain operates multiple store branches across different cities. The dataset includes information about each transaction, such as the branch location, product line, customer gender, payment method, and customer rating. By analyzing this data, the business can uncover patterns and trends that can inform strategic decisions around product assortment, pricing, marketing, and customer experience.

Some key business objectives that can be addressed through this analysis include:

- Identifying the most profitable product lines and optimizing the product mix
- Understanding customer preferences and shopping behavior to enhance the customer experience
- Analyzing the impact of factors like payment method, customer demographics, and location on sales and profitability
- Developing predictive models to forecast sales and optimize inventory management
- Identifying opportunities to improve operational efficiency and reduce costs

Supermarket EDA, Prediction, and Data Modelling

I will start this task of Super Market analysis by importing the necessary Python libraries and loading the dataset:

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

In [3]:
```
df = pd.read_csv("supermarket_sales - Sheet1.csv")
df.head()
```

Out[3]:

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

Data Processing

In [4]:
```
missing_values = df.isna().values.sum()
print("# of missing values:", missing_values)

og_num_rows = df.shape[0]
df_no_na = df.dropna()
altered_num_rows = df_no_na.shape[0]
print("# of rows in original df:", og_num_rows)
print("# of rows after dropping missing values:", altered_num_rows)
```

```
# of missing values: 0
# of rows in original df: 1000
# of rows after dropping missing values: 1000
```

In [5]:
```
df.dtypes
```

Out[5]:
```
Invoice ID                 object
Branch                     object
City                       object
Customer type              object
Gender                     object
Product line               object
Unit price                float64
Quantity                    int64
Tax 5%                    float64
Total                     float64
Date                       object
Time                       object
Payment                    object
cogs                      float64
gross margin percentage   float64
gross income              float64
Rating                    float64
dtype: object
```

In [6]:
```
df.columns
```

Out[6]:
```
Index(['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'],
      dtype='object')
```

In [7]:
```
df.describe()
```

Out[7]:

	Unit price	Quantity	Tax 5%	Total	cogs	gross margin percentage	gross income	10
count	1000.000000	1000.000000	1000.000000	1000.000000	1000.00000	1.000000e+03	1000.000000	
mean	55.672130	5.510000	15.379369	322.966749	307.58738	4.761905e+00	15.379369	
std	26.494628	2.923431	11.708825	245.885335	234.17651	6.131498e-14	11.708825	
min	10.080000	1.000000	0.508500	10.678500	10.17000	4.761905e+00	0.508500	
25%	32.875000	3.000000	5.924875	124.422375	118.49750	4.761905e+00	5.924875	
50%	55.230000	5.000000	12.088000	253.848000	241.76000	4.761905e+00	12.088000	
75%	77.935000	8.000000	22.445250	471.350250	448.90500	4.761905e+00	22.445250	
max	99.960000	10.000000	49.650000	1042.650000	993.00000	4.761905e+00	49.650000	

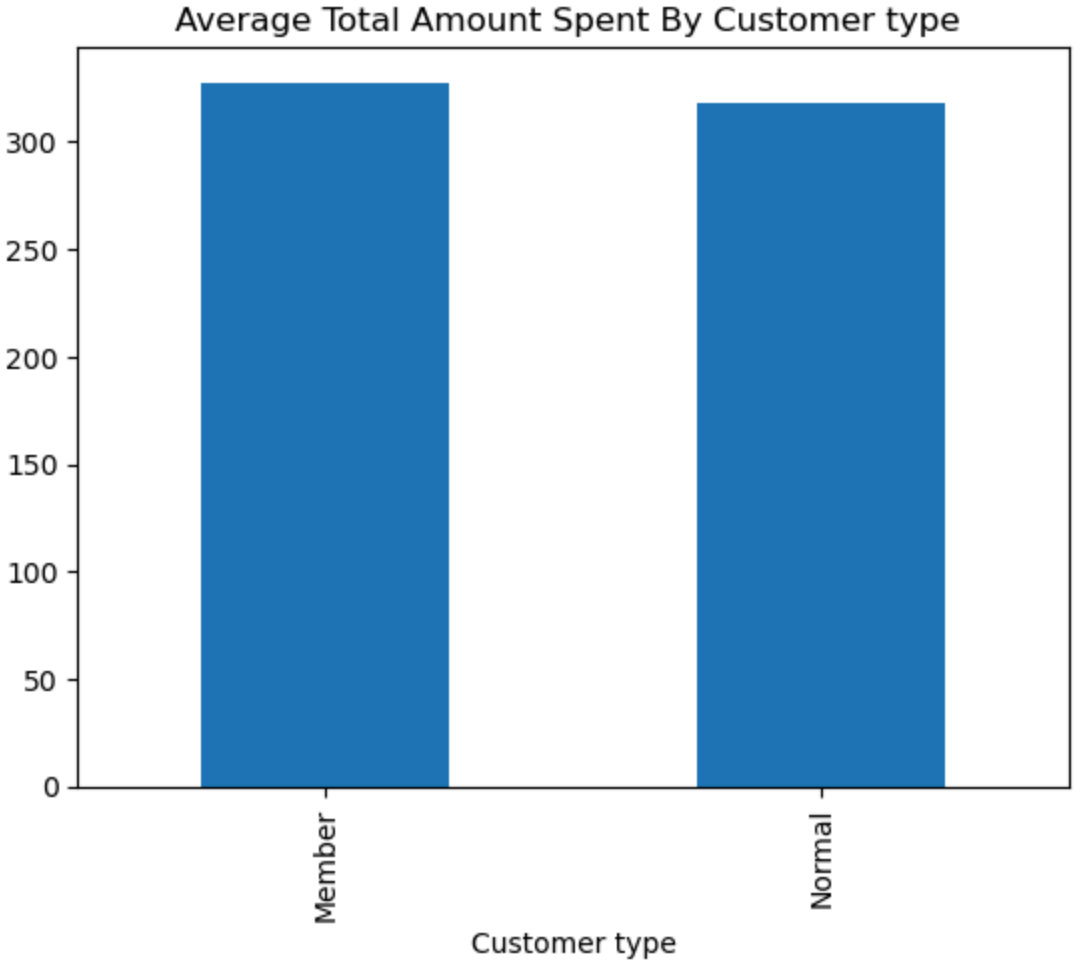
Exploratory Data Analysis (EDA)

1. What is the average total amount spent by each customer type and gender

In [8]:
```
df_customer_type = df.groupby('Customer type')['Total'].mean()
print(df_customer_type)

df_customer_type.plot(kind='bar', title = 'Average Total Amount Spent By Customer type
```

```
Customer type
Member    327.791305
Normal    318.122856
Name: Total, dtype: float64
```

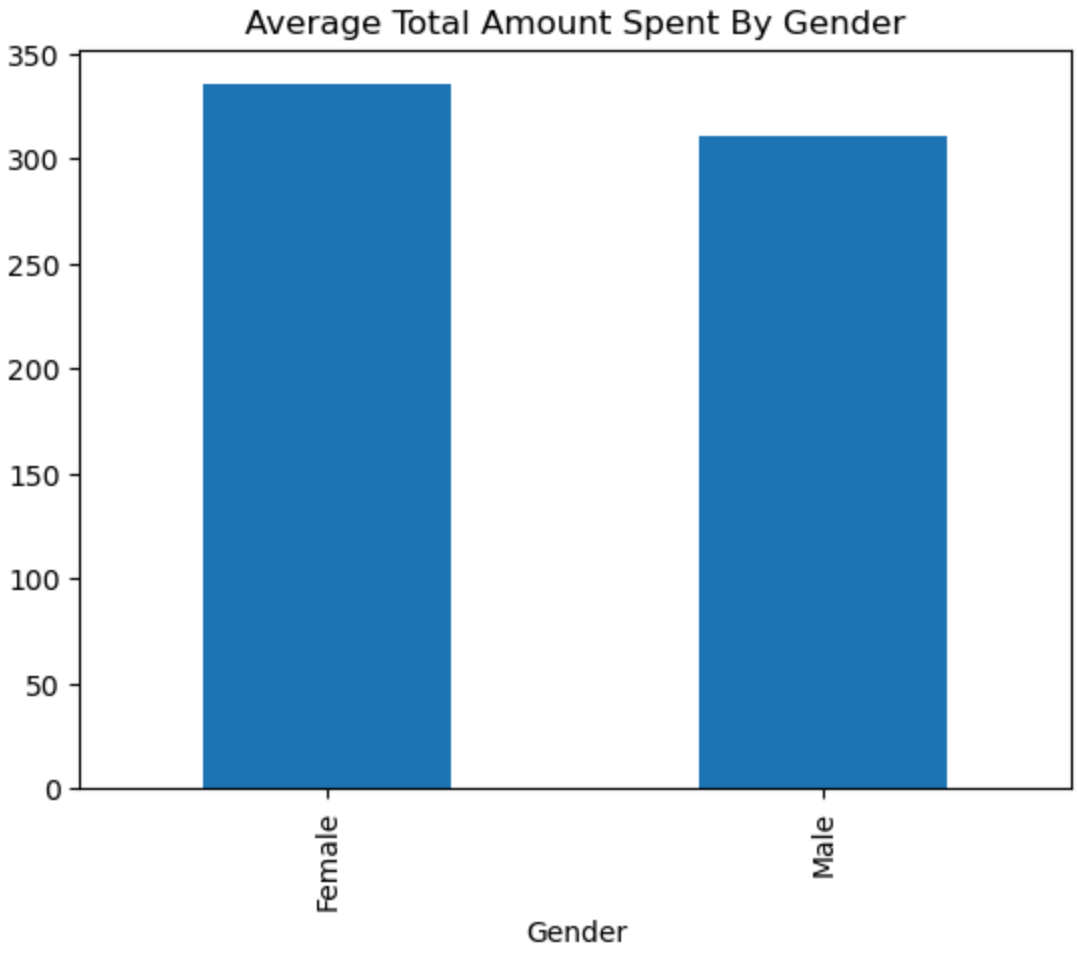
Out[8]:
```
<Axes: title={'center': 'Average Total Amount Spent By Customer type '}, xlabel='Cust
omer type'>
```



In [9]:
```
df_genderr = df.groupby('Gender')['Total'].mean()
print(df_genderr)

df_genderr.plot(kind='bar', title = 'Average Total Amount Spent By Gender')
```

```
Gender
Female    335.095659
Male      310.789226
Name: Total, dtype: float64
```

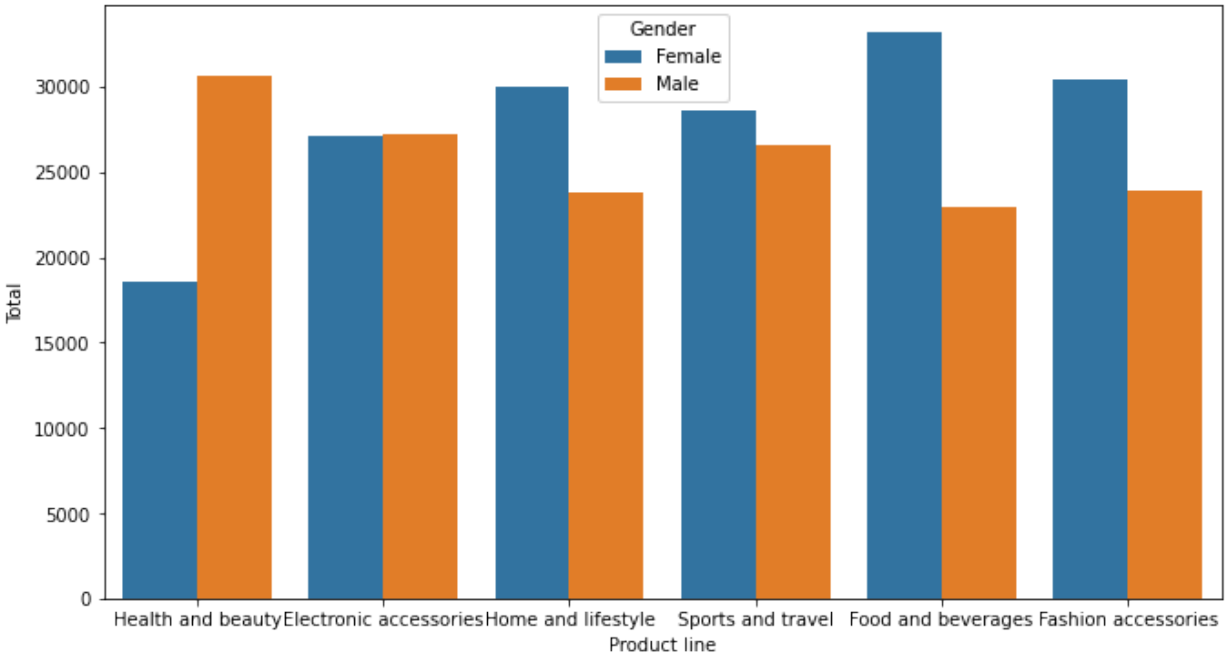
Out[9]:
```
<Axes: title={'center': 'Average Total Amount Spent By Gender'}, xlabel='Gender'>
```



2. What is the total sales for each gender and product line combination?

In [9]:
```
#Gender vs. Product Line Combination
plt.figure(figsize=(11,6))
sns.barplot(x='Product line', y = 'Total', hue = 'Gender', data = df, estimator = sum,
```

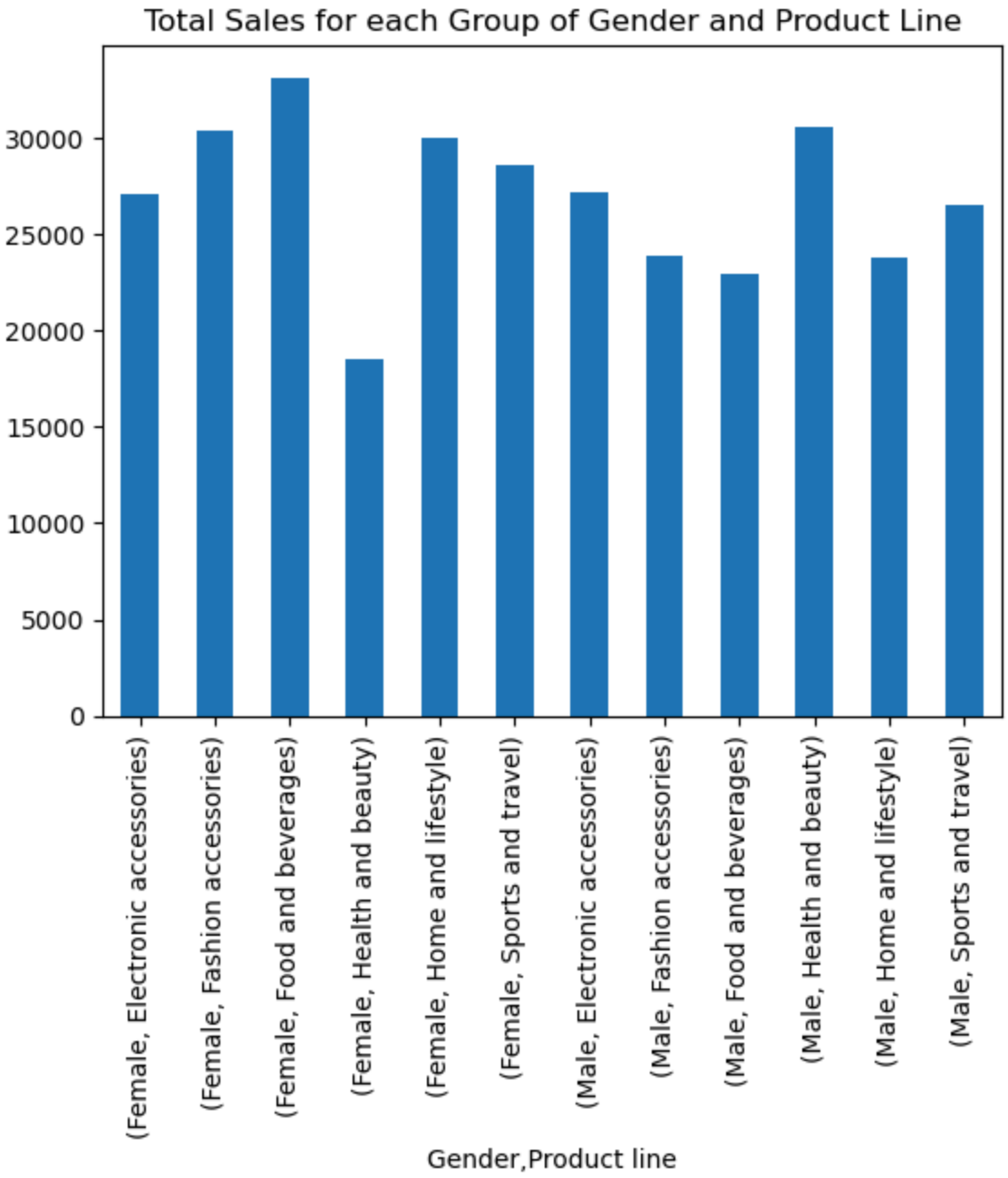
Out[9]:
```
<AxesSubplot:xlabel='Product line', ylabel='Total'>
```



In [11]:
```
#group data by gender and product line and calculate the total sales for each group

df_gender_product_line = df.groupby(['Gender', 'Product line'])['Total'].sum()
print(df_gender_product_line)

df_gender_product_line.plot(kind='bar', title = 'Total Sales for each Group of Gender
```

```
Gender  Product line
Female  Electronic accessories    27102.0225
        Fashion accessories       30437.4000
        Food and beverages        33170.9175
        Health and beauty         18560.9865
        Home and lifestyle        30036.8775
        Sports and travel         28574.7210
Male    Electronic accessories    27235.5090
        Fashion accessories       23868.4950
        Food and beverages        22973.9265
        Health and beauty         30632.7525
        Home and lifestyle        23825.0355
        Sports and travel         26548.1055
Name: Total, dtype: float64
```

Out[11]:
```
<Axes: title={'center': 'Total Sales for each Group of Gender and Product Line'}, xla
bel='Gender,Product line'>
```

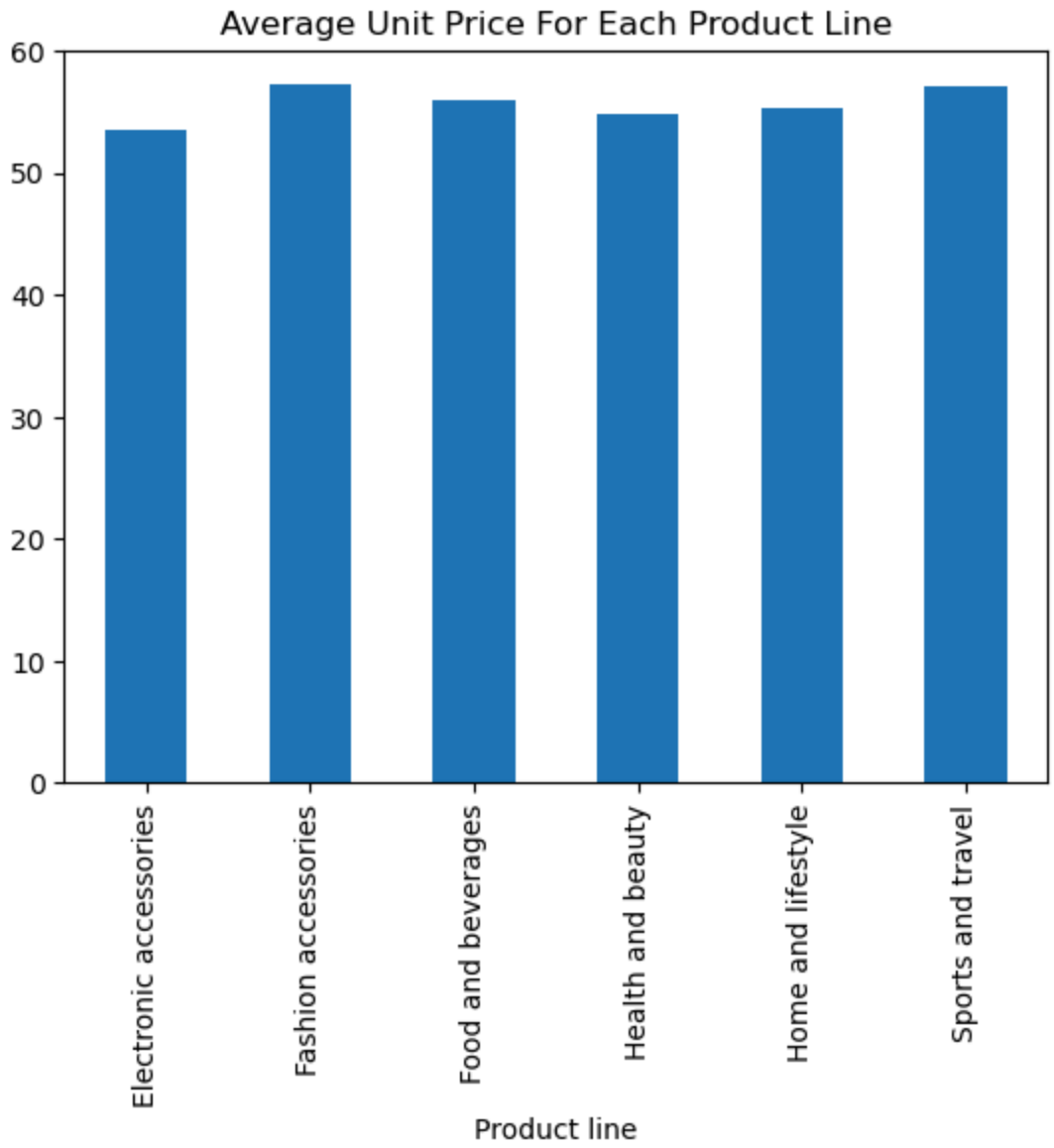


3. What is the average unit price for each product line?

In [10]:
```
#calculate the average unit price for each product line

df_product_line_price = df.groupby('Product line')['Unit price'].mean()
print(df_product_line_price)

df_product_line_price.plot(kind='bar', title = 'Average Unit Price For Each Product Li
```

```
Product line
Electronic accessories    53.551588
Fashion accessories       57.153652
Food and beverages        56.008851
Health and beauty         54.854474
Home and lifestyle        55.316937
Sports and travel         56.993253
Name: Unit price, dtype: float64
```

Out[10]: <Axes: title={'center': 'Average Unit Price For Each Product Line'}, xlabel='Product line'>



4. What is the overall gross margin percentage?

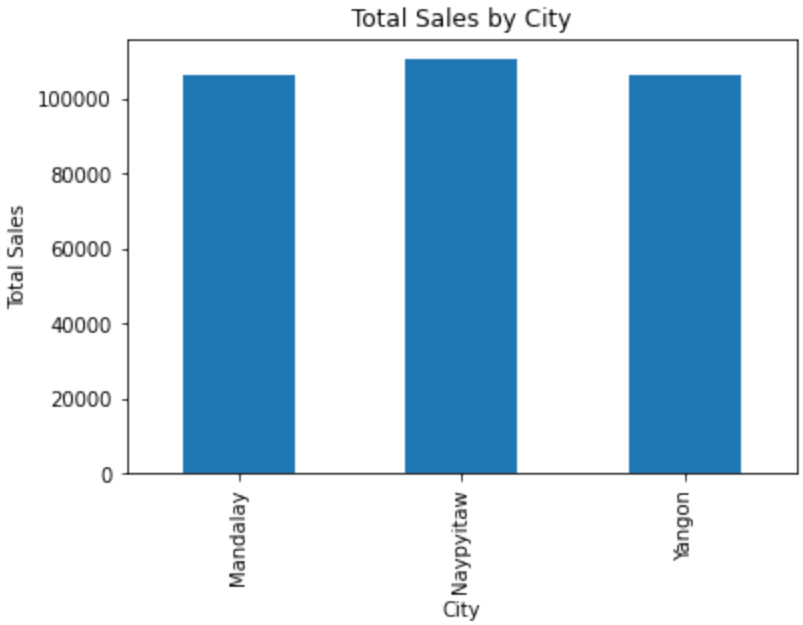
In [12]:
```
#calculate the overall gross margin percentage

df['gross_margin'] = (df['Total'] - df['cogs']) / df['Total']
overall_gross_margin = df['gross_margin'].mean()

print(overall_gross_margin)
```

```
0.04761904761904762
```

A gross margin of **0.047** means that out of the total revenue, only **4.76%** is left after accounting for the COGS. This indicates a low level of profitability for the Supermarket. To improve the profitability, the business can try to reduce the COGS or increase the revenue.

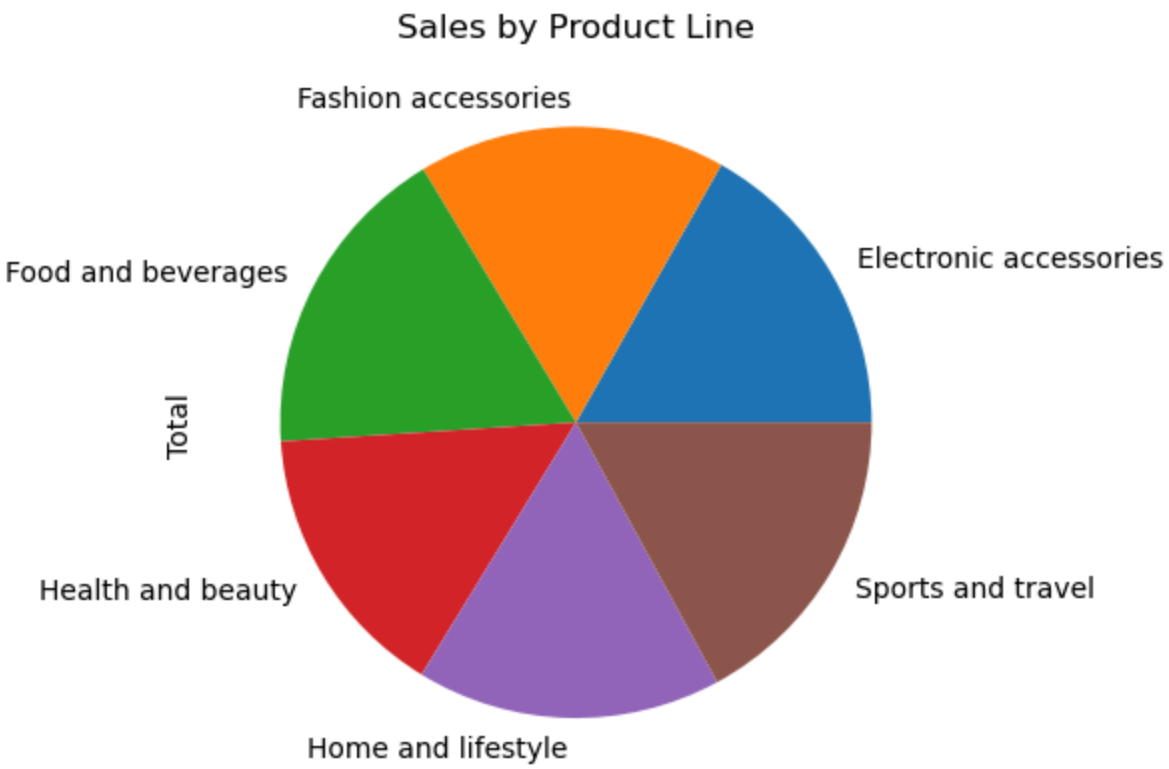
5. Which cities are the biggest contributors to the overall sales?

In [13]:
```
import matplotlib.pyplot as plt

#calculate the total sales for each city
df_city_sales = df.groupby('City')['Total'].sum()

df_city_sales.plot(kind='bar')

plt.title('Total Sales by City')
plt.xlabel('City')
plt.ylabel('Total Sales')

plt.show()
```



6. Which product lines are the most popular among customers?

In [15]:
```
#pie chart to visualize the distribution of sales across different product lines

df.groupby('Product line')['Total'].sum().plot(kind='pie')
plt.title('Sales by Product Line')
plt.show()
```

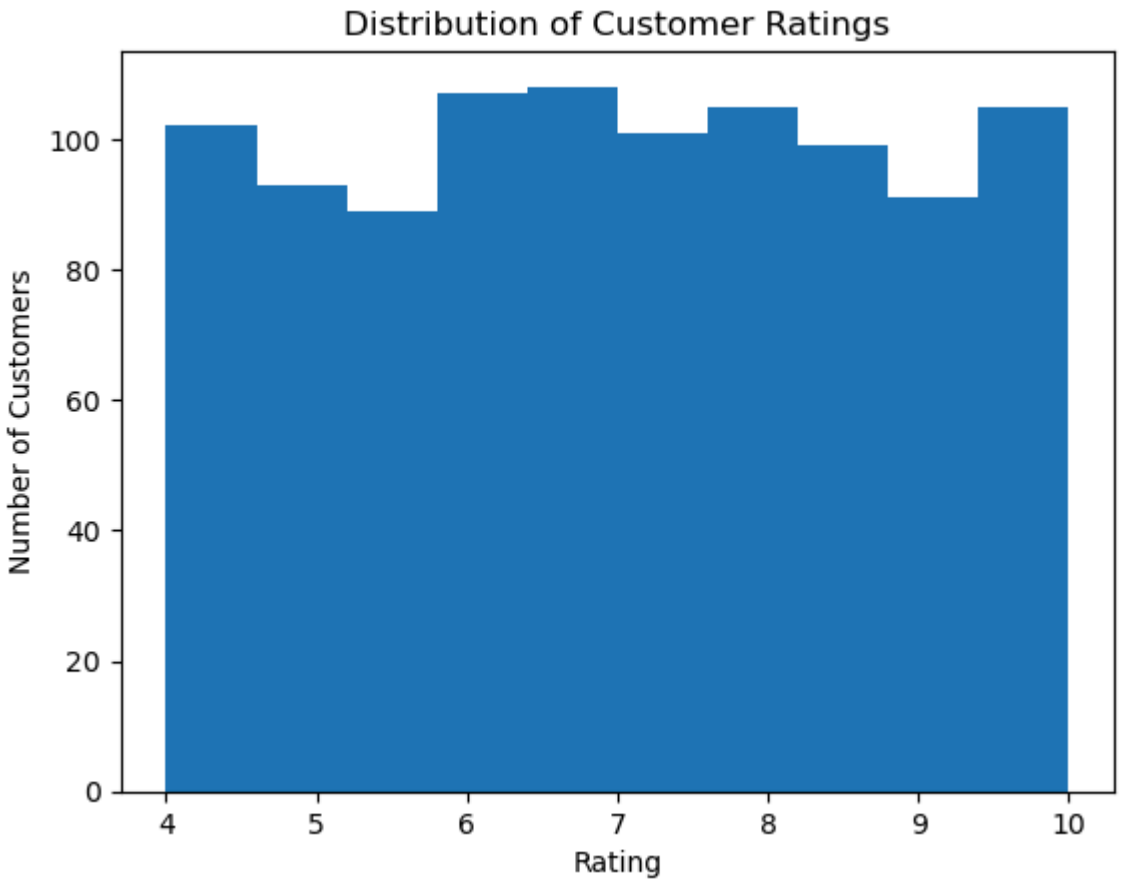


7. What is the relationship between the unit price and the quantity of each product?

We can see a correlation value of **0.010778** indicates a very weak positive correlation between the two variables. This means that there is a very weak relationship between the two variables and as one variable increases, the other variable also increases, but only slightly.

8. What is the overall satisfaction level of customers with the supermarket?

In [20]:
```
#histogram to visualize the distribution of customer ratings

df['Rating'].plot(kind='hist')

plt.title('Distribution of Customer Ratings')
plt.xlabel('Rating')
plt.ylabel('Number of Customers')

plt.show()
```


Distribution of Customer Ratings
Number of Customers
0
20
40
60
80
100
4
5
6
7
8
9
10
Rating

9. Which branches are performing well in terms of gross margin percentage?

In [28]:
```
import matplotlib.pyplot as plt

#calculate the gross margin percentage for each branch
df_branch_margin = df.groupby('Branch')['gross margin percentage'].mean()
```

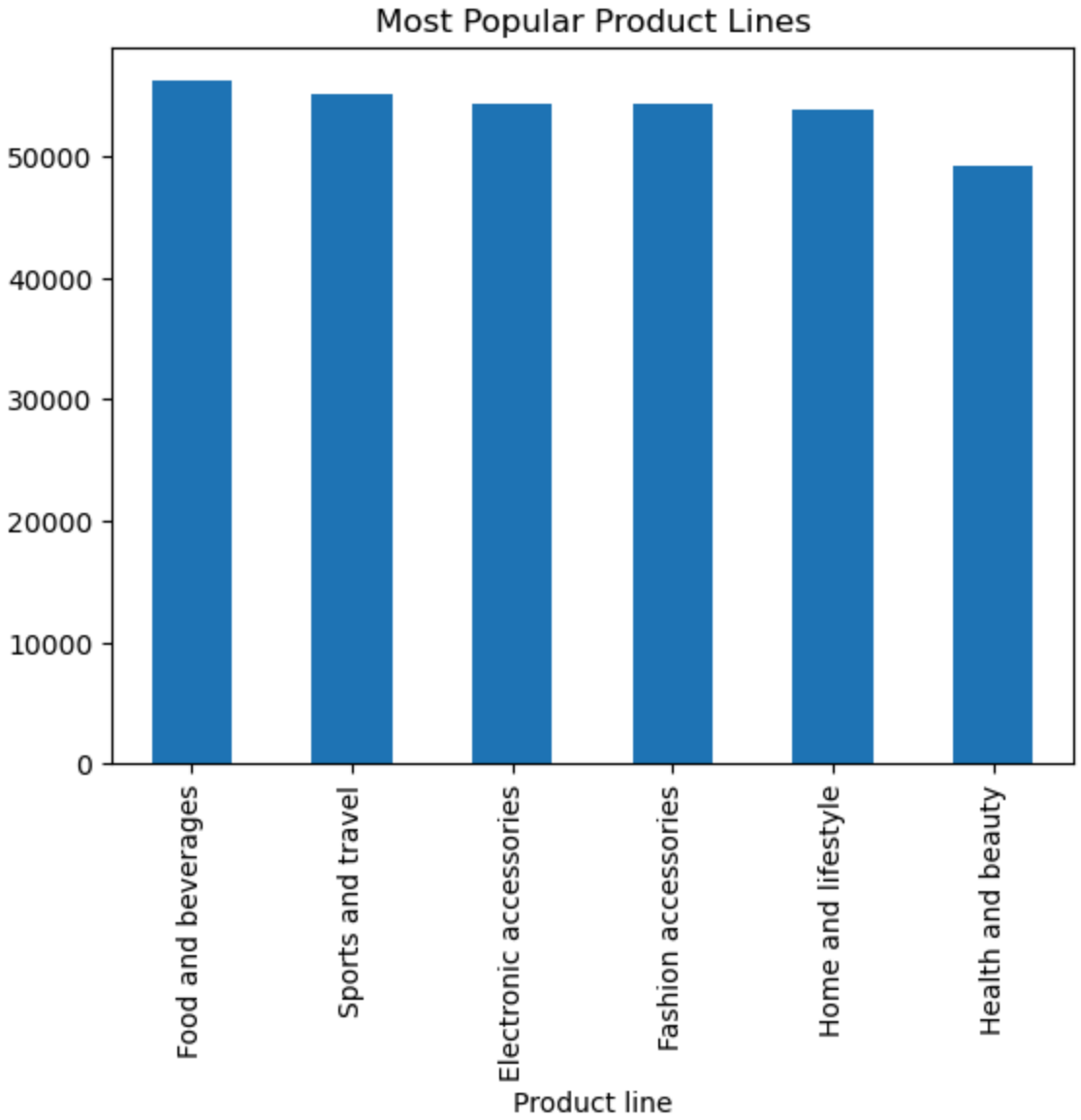
As stated earlier, a gross margin of **0.047** means that out of the total revenue, only 4.76% is left after accounting for the COGS. This indicates a low level of profitability for the Supermarket. To improve the profitability, the business can try to reduce the COGS or increase the revenue.

10. What are the most popular product lines in the supermarket?

In [30]:
```
#group the data by product line and calculate total sales for each product line
product_line_sales = df.groupby('Product line')['Total'].sum()

#sort the product lines by total sales in descending order
product_line_sales = product_line_sales.sort_values(ascending=False)

print('Most popular product lines:')
print(product_line_sales.head(10))

product_line_sales.head(10).plot(kind='bar', title = 'Most Popular Product Lines')
```

```
Most popular product lines:
Product line
Food and beverages        56144.8440
Sports and travel         55122.8265
Electronic accessories    54337.5315
Fashion accessories       54305.8950
Home and lifestyle        53861.9130
Health and beauty         49193.7390
Name: Total, dtype: float64
```

Out[30]: <Axes: title={'center': 'Most Popular Product Lines'}, xlabel='Product line'>



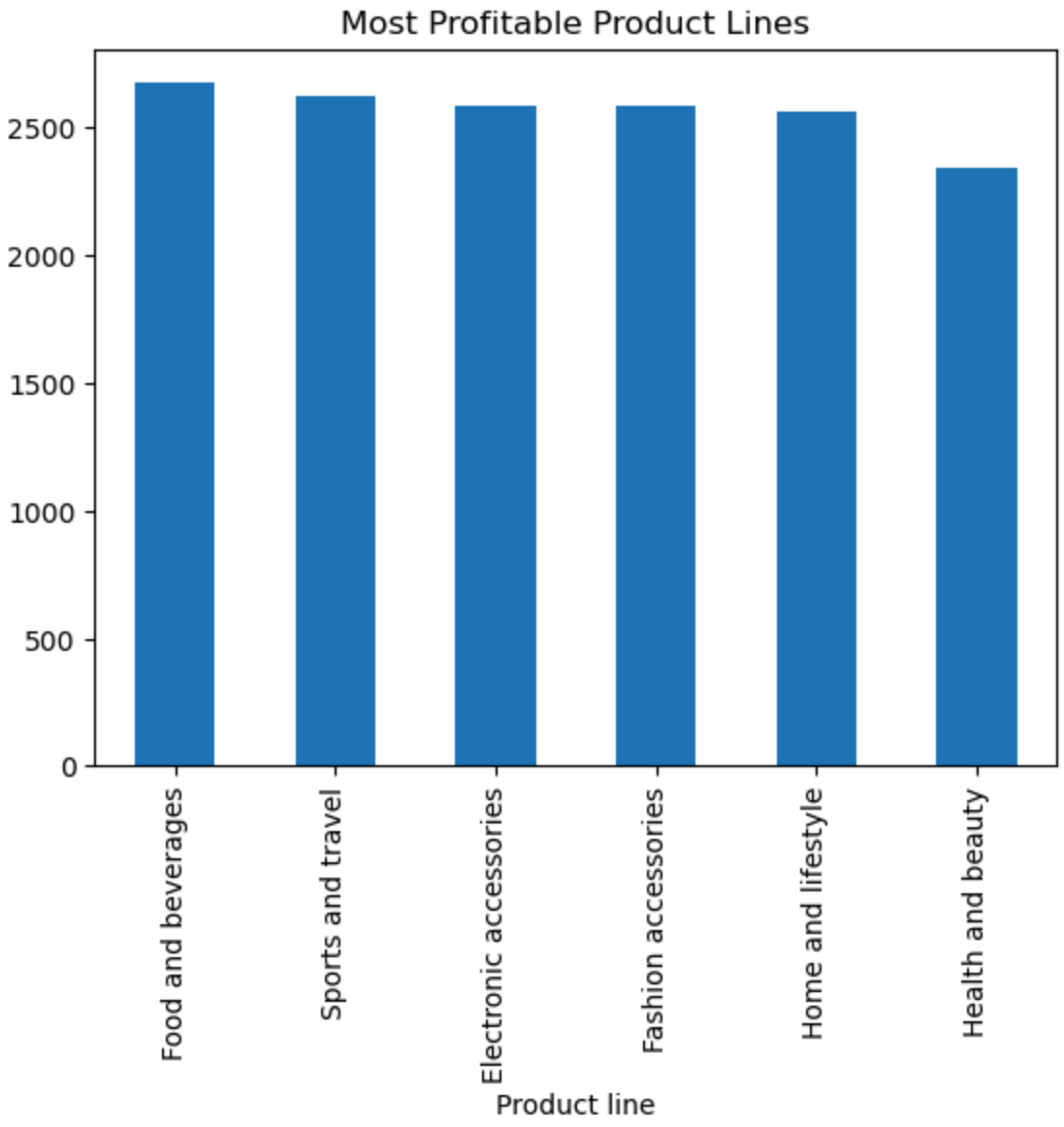
11. What are the most profitable product lines in the supermarket?

In [32]:
```
#group data by product line and calculate the total gross income for each product line
product_line_income = df.groupby('Product line')['gross income'].sum()

#sort product lines by total gross income in descending order
product_line_income = product_line_income.sort_values(ascending=False)

print('Most profitable product lines:')
print(product_line_income.head(10))
```

```
product_line_income.head(10).plot(kind='bar', title = 'Most Profitable Product Lines')
```

```
Most profitable product lines:
Product line
Food and beverages        2673.5640
Sports and travel         2624.8965
Electronic accessories    2587.5015
Fashion accessories       2585.9950
Home and lifestyle        2564.8530
Health and beauty         2342.5590
Name: gross income, dtype: float64
```

Out[32]: <Axes: title={'center': 'Most Profitable Product Lines'}, xlabel='Product line'>



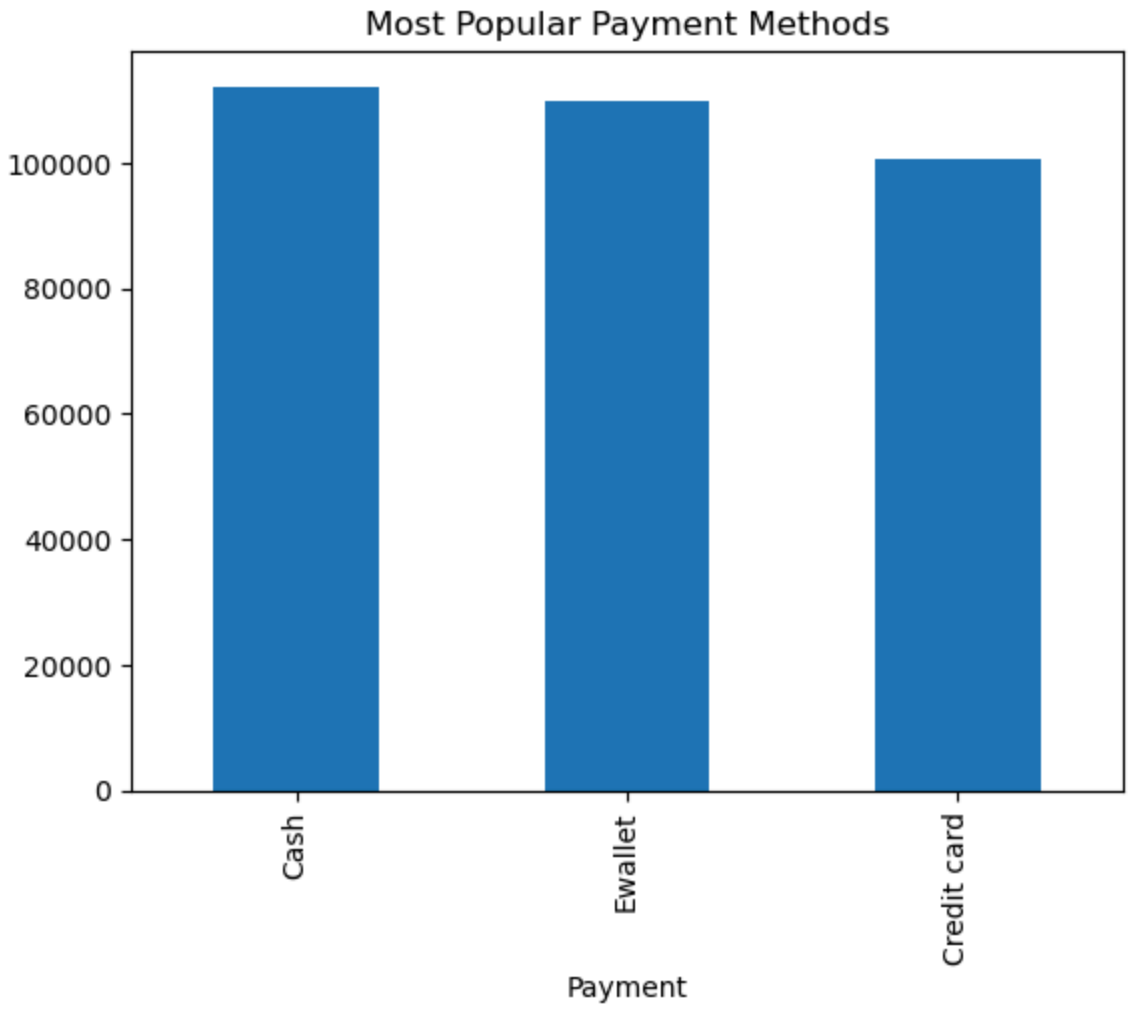
12. What are the most popular payment methods used in the supermarket?

In [34]:
```
#group data by payment method and calculate the total sales for each payment method
payment_method_sales = df.groupby('Payment')['Total'].sum()

#sort payment methods by total sales in descending order
payment_method_sales = payment_method_sales.sort_values(ascending=False)
```

```
print('Most popular payment methods:')
print(payment_method_sales.head(10))

payment_method_sales.head(10).plot(kind='bar', title = 'Most Popular Payment Methods')
```

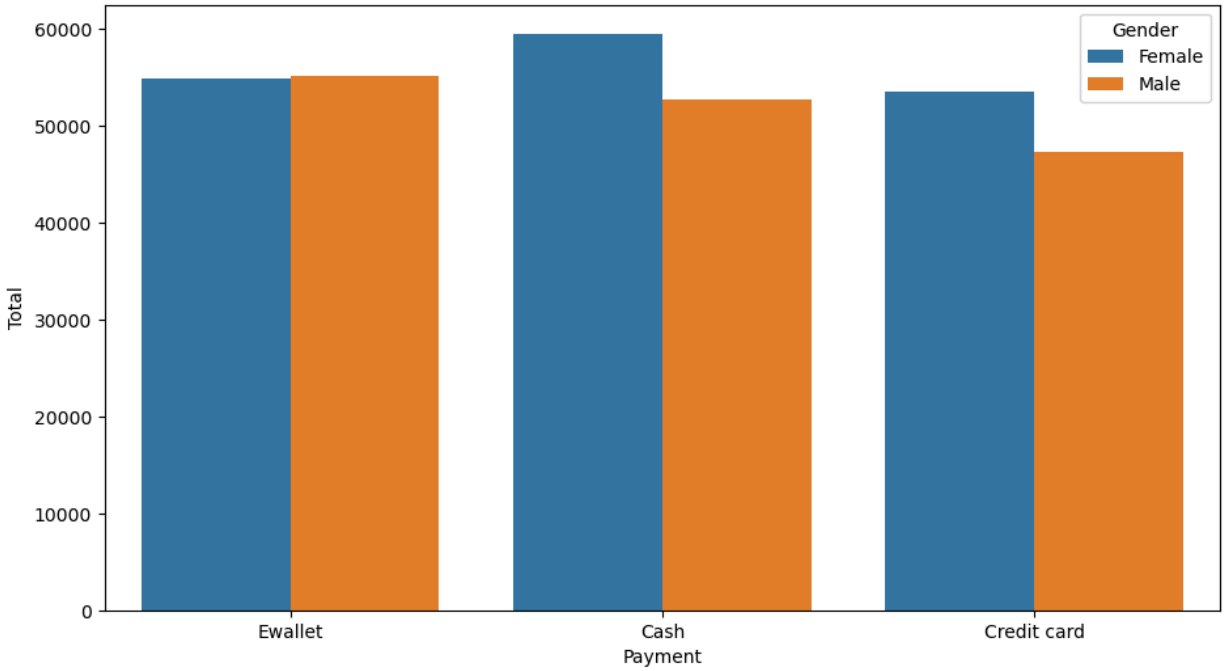
```
Most popular payment methods:
Payment
Cash           112206.570
Ewallet        109993.107
Credit card    100767.072
Name: Total, dtype: float64
```

Out[34]: <Axes: title={'center': 'Most Popular Payment Methods'}, xlabel='Payment'>



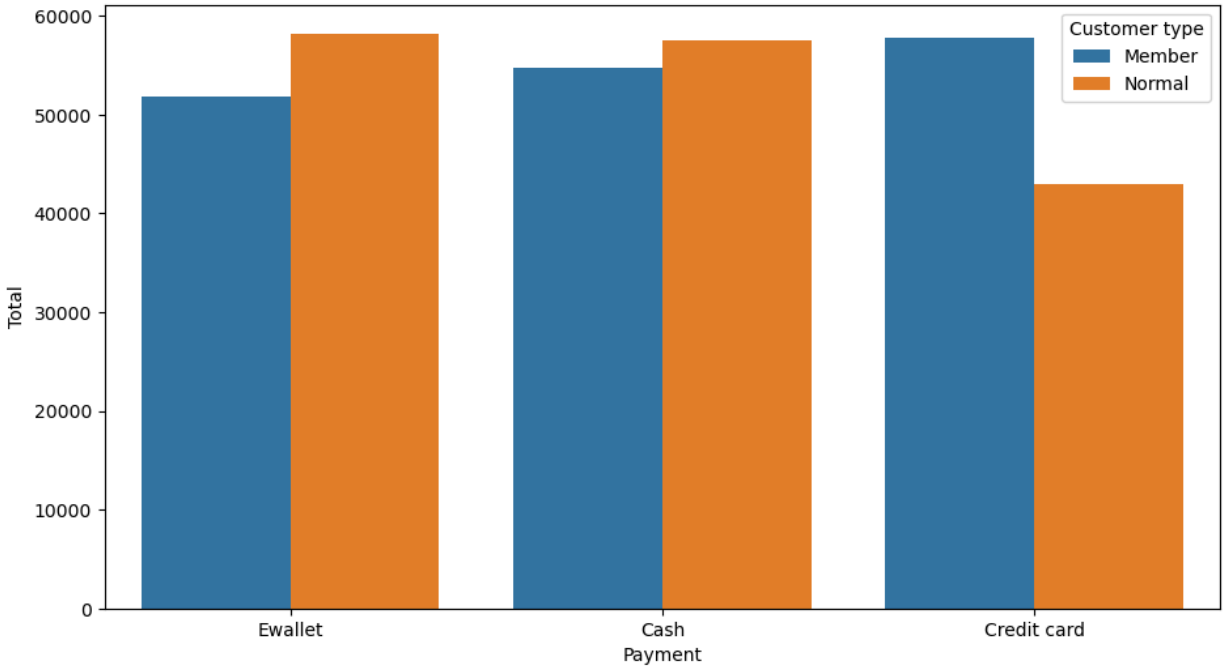
In [38]:
```
#total sales for each gender by payment method

plt.figure(figsize=(11,6))
sns.barplot(x = 'Payment', y = 'Total', hue = 'Gender', data = df, errorbar=None, esti
```

Out[38]: <Axes: xlabel='Payment', ylabel='Total'>



In [41]:
```
#total sales for each customer type by payment method

plt.figure(figsize=(11,6))
sns.barplot(x = 'Payment', y = 'Total', hue = 'Customer type', data = df, errorbar=Nor
```

Out[41]:
```
<Axes: xlabel='Payment', ylabel='Total'>
```



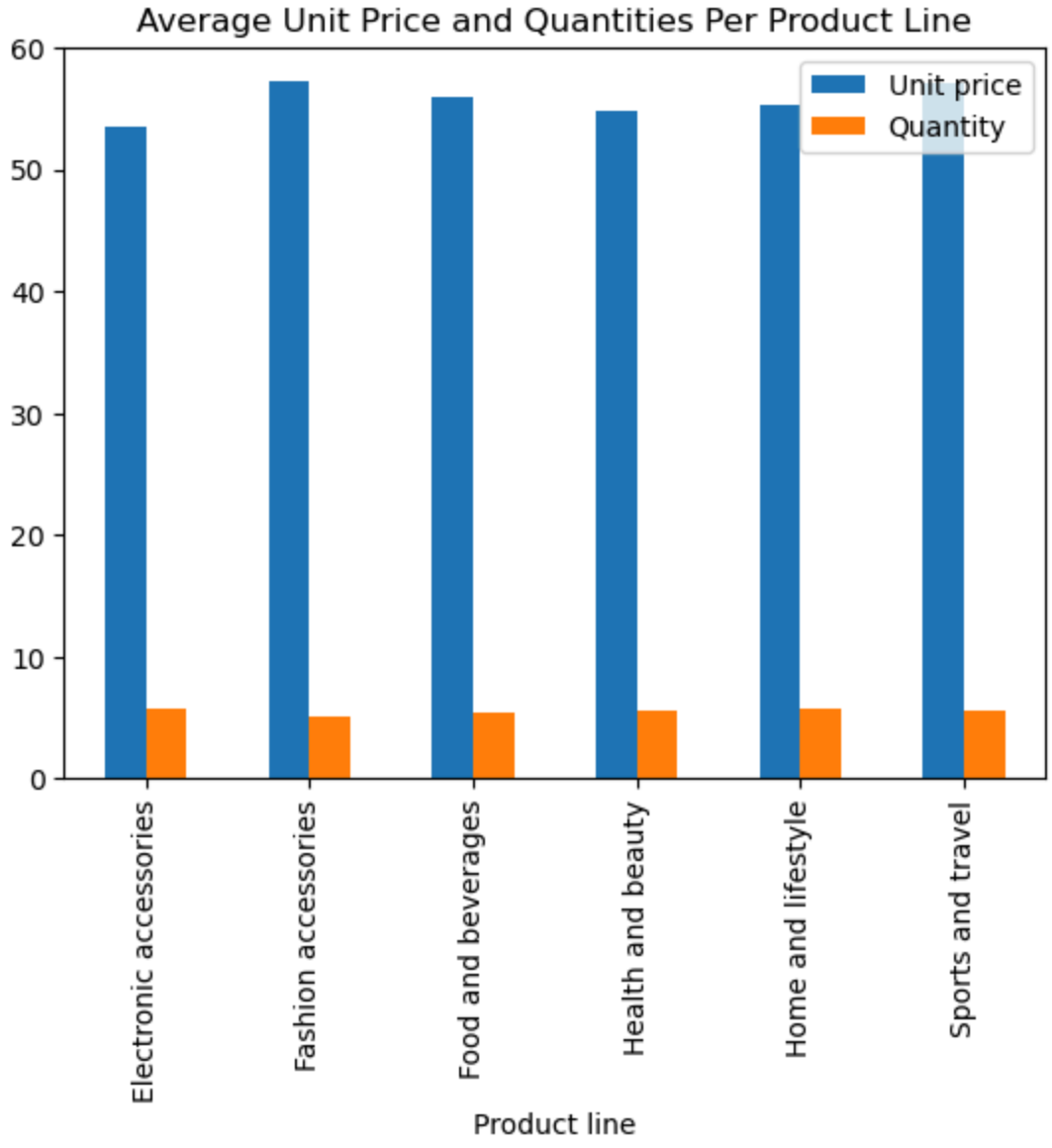
13. What are the average unit prices and quantities sold for each product line?

In [44]:
```
#group data by product line and calculate the average unit price and quantity for each
product_line_data = df.groupby('Product line')[['Unit price', 'Quantity']].mean()

print('Average unit prices and quantities for each product line:')
print(product_line_data)
```

```
product_line_data.head(10).plot(kind='bar', title='Average Unit Price and Quantities P
```

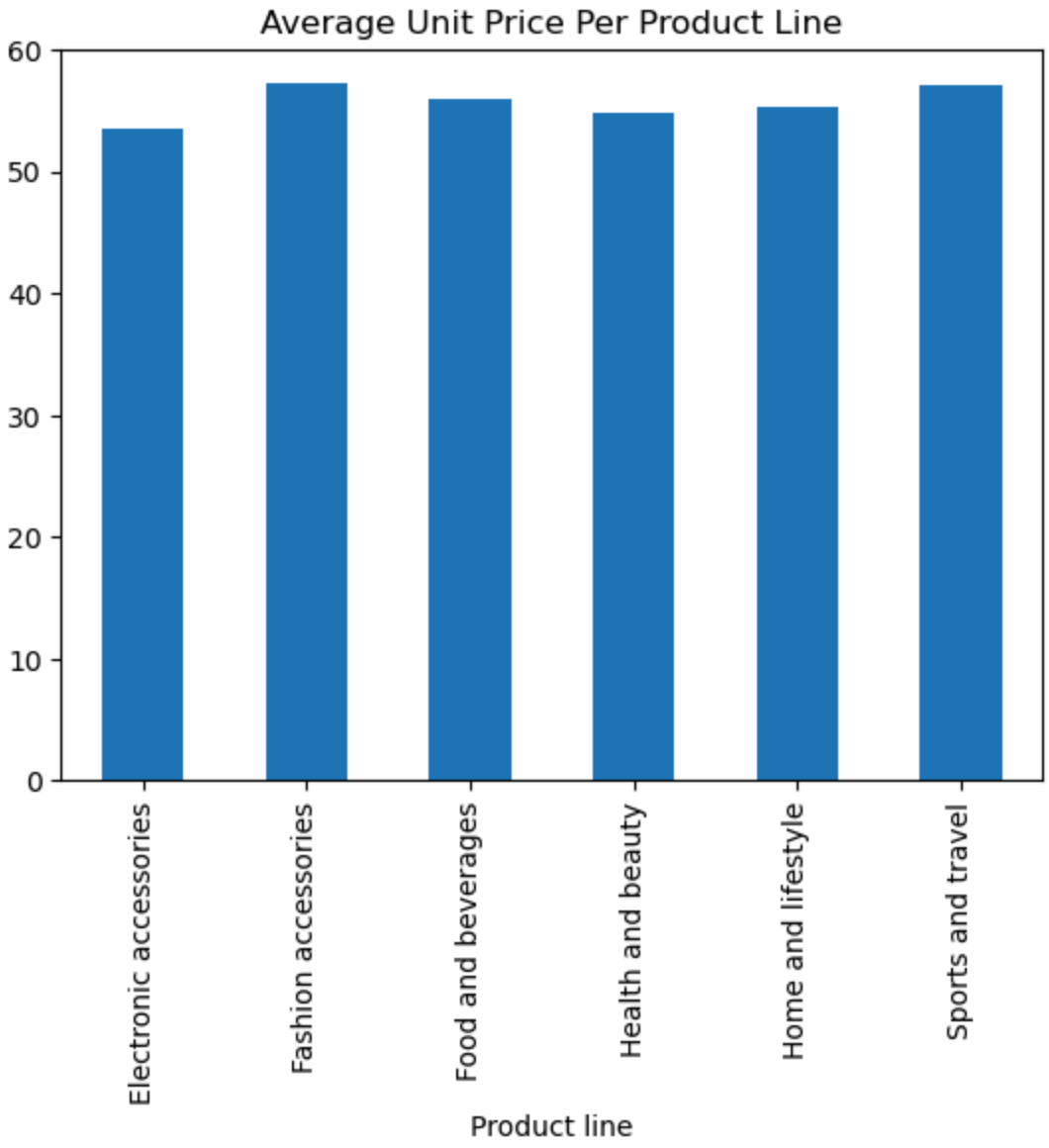
```
Average unit prices and quantities for each product line:
                        Unit price  Quantity
Product line
Electronic accessories   53.551588  5.711765
Fashion accessories      57.153652  5.067416
Food and beverages       56.008851  5.471264
Health and beauty        54.854474  5.618421
Home and lifestyle       55.316937  5.693750
Sports and travel        56.993253  5.542169
```

Out[44]:
```
<Axes: title={'center': 'Average Unit Price and Quantities Per Product Line'}, xlabel
='Product line'>
```



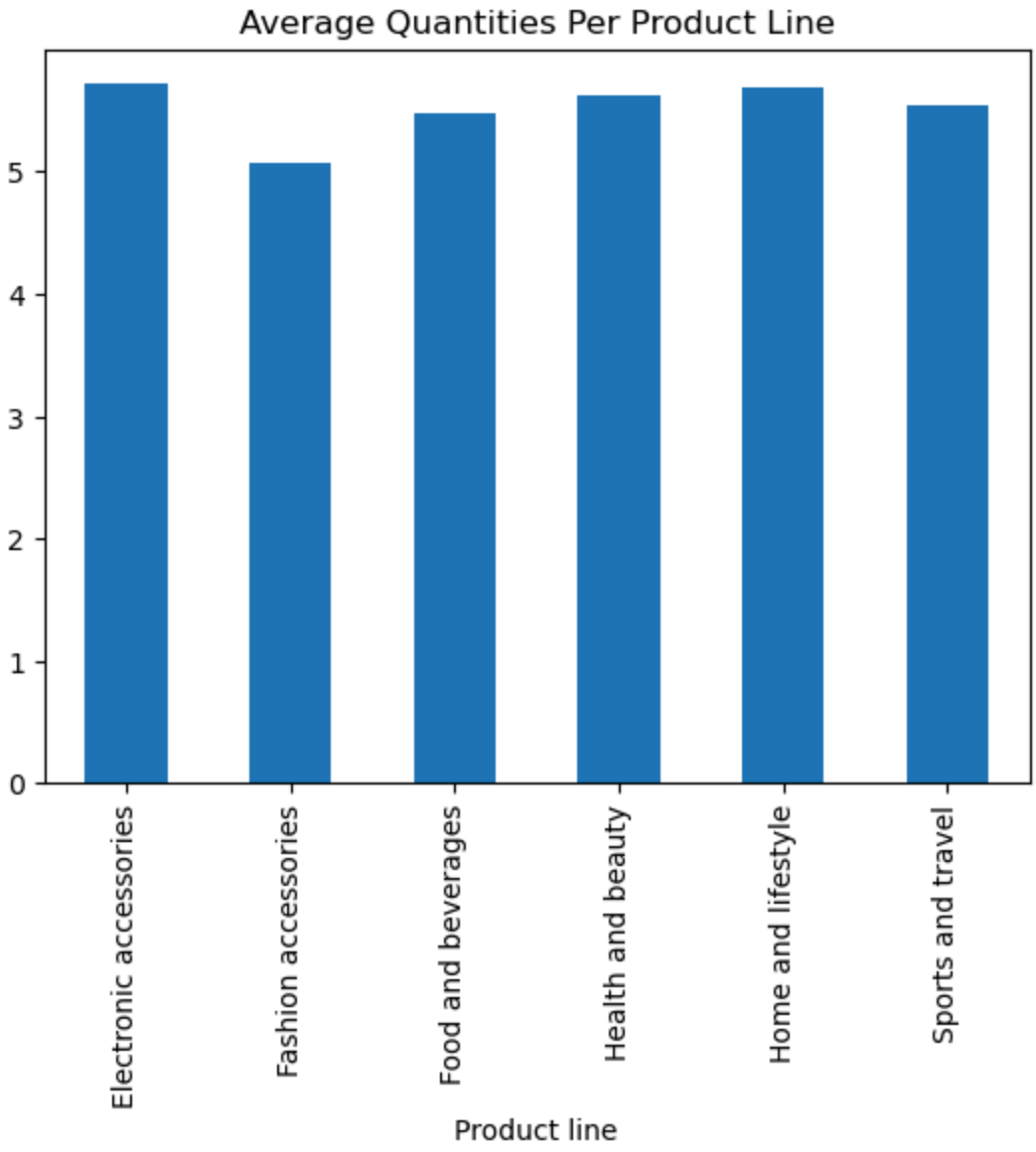
In [45]:
```
#visualize the average unit prices for each product line
product_line_data['Unit price'].plot(kind='bar', title = 'Average Unit Price Per Produ
```

Out[45]:
```
<Axes: title={'center': 'Average Unit Price Per Product Line'}, xlabel='Product lin
e'>
```



In [46]:
```
#visualize the average quantities for each product line
product_line_data['Quantity'].plot(kind='bar', title = 'Average Quantities Per Product
```

Out[46]:
```
<Axes: title={'center': 'Average Quantities Per Product Line'}, xlabel='Product lin
e'>
```



14. What are the average gross margins and gross incomes for each product line?

In [48]:
```
#group data by product line and calculate the average gross margin and gross income fo
product_line_dataa = df.groupby('Product line')[['gross margin percentage', 'gross inc

print('Average gross margins and gross incomes for each product line:')
print(product_line_dataa)
```

```
Average gross margins and gross incomes for each product line:
                         gross margin percentage  gross income
Product line
Electronic accessories                  4.761905     15.220597
Fashion accessories                     4.761905     14.528062
Food and beverages                      4.761905     15.365310
Health and beauty                       4.761905     15.411572
Home and lifestyle                      4.761905     16.030331
Sports and travel                       4.761905     15.812630
```

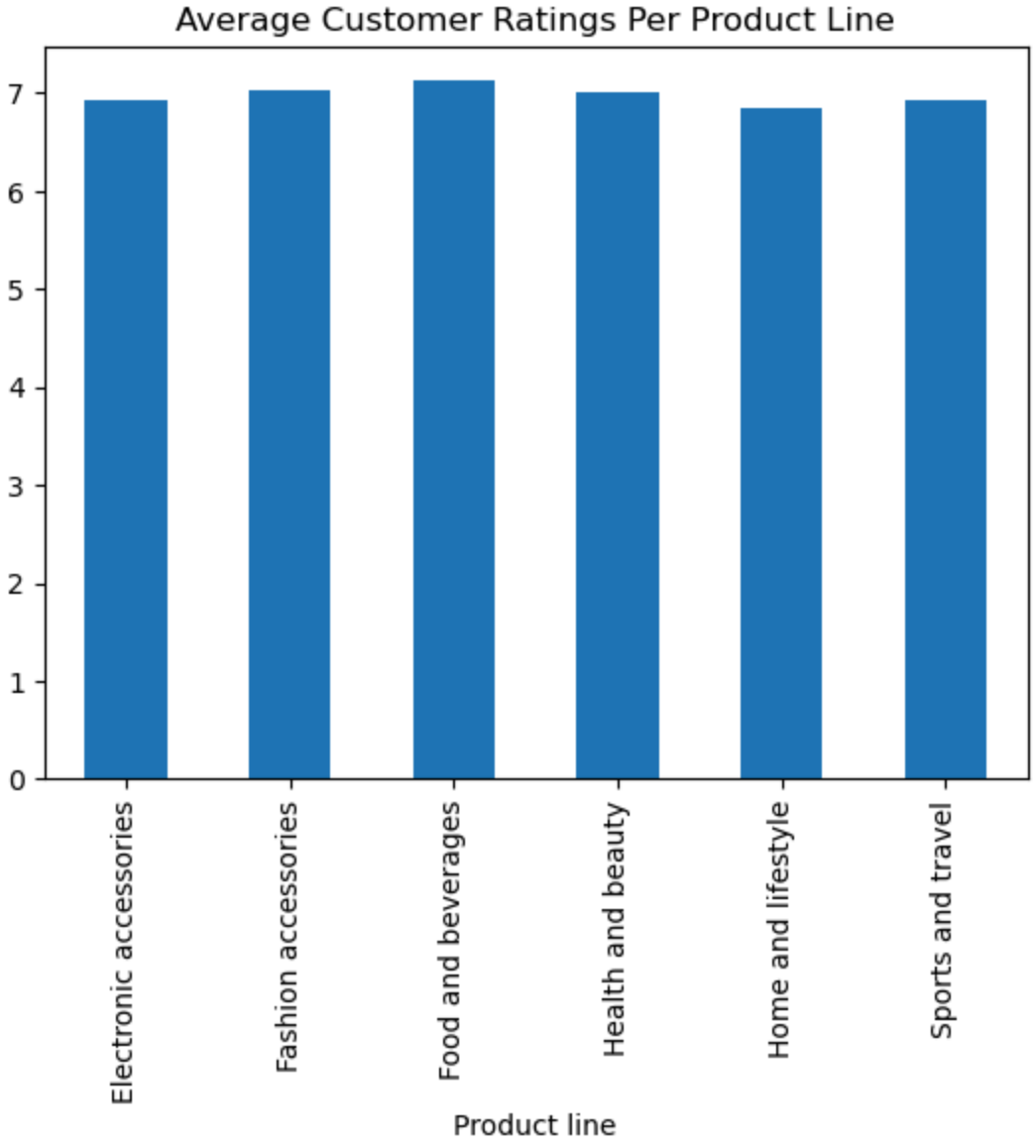
15. What are the average customer ratings for each product line?

In [50]:
```
#group data by product line and calculate the average customer rating for each product
product_line_datta = df.groupby('Product line')['Rating'].mean()
```

```
print('Average customer ratings for each product line:')
print(product_line_datta)

product_line_datta.head(10).plot(kind='bar', title = 'Average Customer Ratings Per Prc
```

```
Average customer ratings for each product line:
Product line
Electronic accessories    6.924706
Fashion accessories       7.029213
Food and beverages        7.113218
Health and beauty         7.003289
Home and lifestyle        6.837500
Sports and travel         6.916265
Name: Rating, dtype: float64
```

Out[50]: <Axes: title={'center': 'Average Customer Ratings Per Product Line'}, xlabel='Product line'>



16. What are the most popular branches of the supermarket in terms of sales and customer ratings?

In [52]:
```
#group data by branch and calculate the total sales and average customer rating for ea
branch_data = df.groupby('Branch')[['Total', 'Rating']].agg(['sum', 'mean'])
```

```
branch_data = branch_data.sort_values(('Total', 'sum'), ascending=False)

print(branch_data)
```

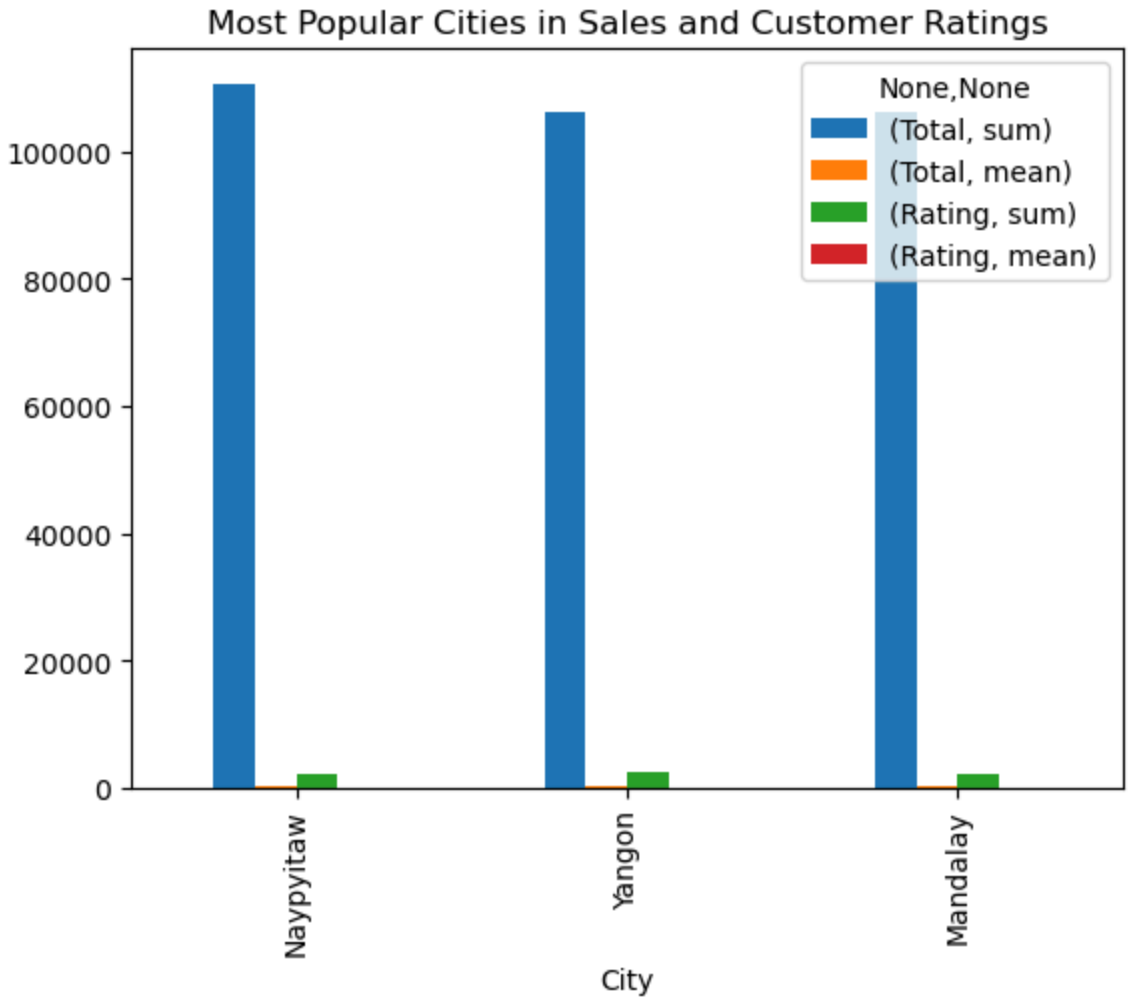
```
              Total              Rating
                sum        mean     sum      mean
Branch
C       110568.7065  337.099715  2319.9  7.072866
A       106200.3705  312.354031  2389.2  7.027059
B       106197.6720  319.872506  2263.6  6.818072
```

17. What are the most popular cities for the supermarket in terms of sales and customer ratings?

In [55]:
```
#group data by city and calculate the total sales and average customer rating for each
city_data = df.groupby('City')[['Total', 'Rating']].agg(['sum', 'mean'])

#sort cities by total sales in descending order
city_data = city_data.sort_values(('Total', 'sum'), ascending=False)

print('Most popular cities:')
print(city_data.head(10))

city_data.head(10).plot(kind='bar', title='Most Popular Cities in Sales and Customer R
```

```
Most popular cities:
                Total              Rating
                  sum        mean     sum      mean
City
Naypyitaw  110568.7065  337.099715  2319.9  7.072866
Yangon     106200.3705  312.354031  2389.2  7.027059
Mandalay   106197.6720  319.872506  2263.6  6.818072
```

Out[55]:
```
<Axes: title={'center': 'Most Popular Cities in Sales and Customer Ratings'}, xlabel
='City'>
```

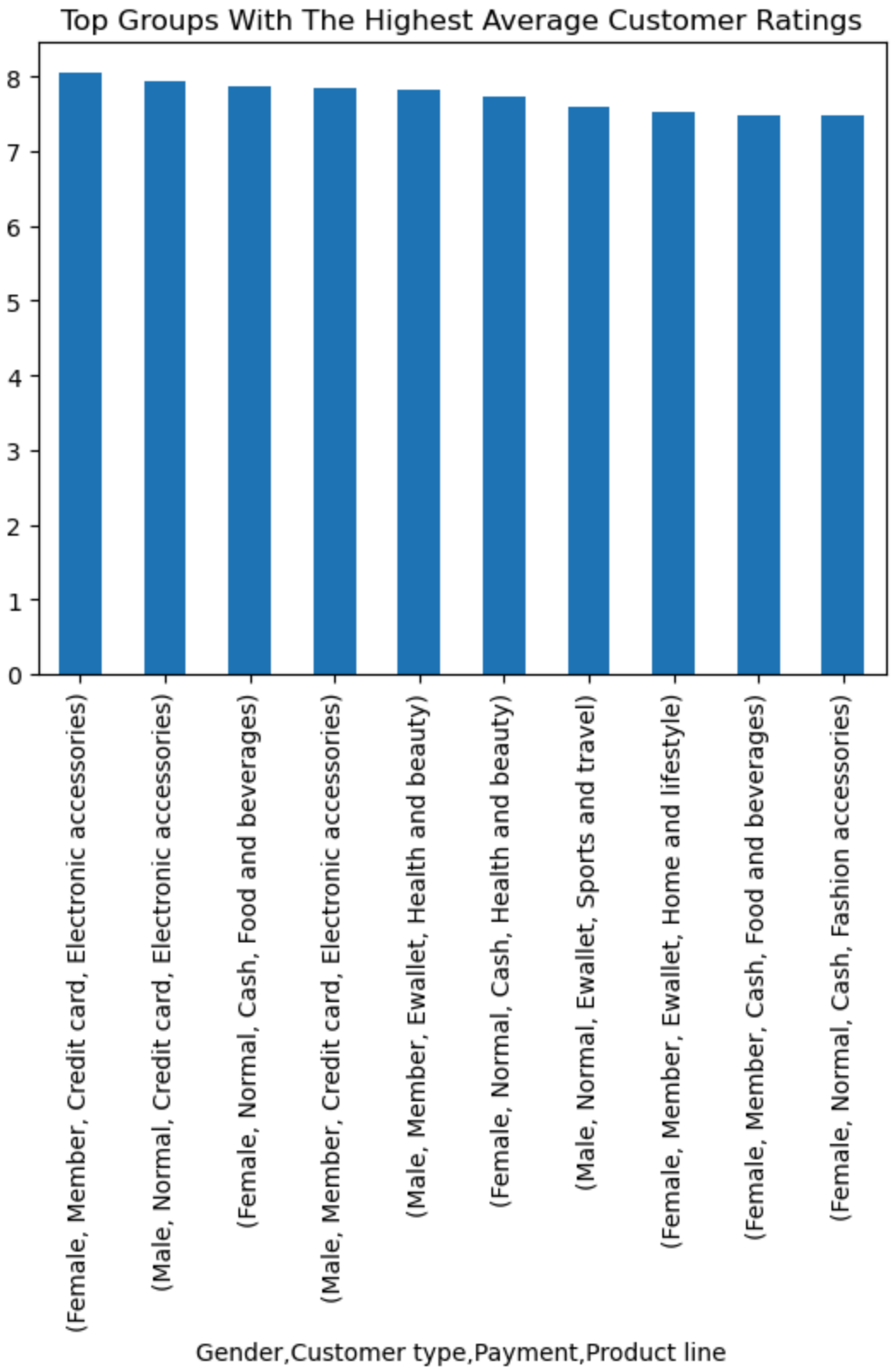


18. What are the characteristics of the most satisfied customers in terms of gender, customer type, payment method, and product line?

In [57]:
```
#group data by gender, customer type, payment method, and product line and calculate t
customer_data = df.groupby(['Gender', 'Customer type', 'Payment', 'Product line'])['Ra

#sort groups by average customer rating in descending order
customer_data = customer_data.sort_values(ascending=False)

print('Top 10 groups with the highest average customer ratings:')
print(customer_data.head(10))

customer_data.head(10).plot(kind='bar', title = 'Top Groups With The Highest Average C
```

```
Top 10 groups with the highest average customer ratings:
Gender  Customer type  Payment      Product line
Female  Member         Credit card  Electronic accessories    8.050000
Male    Normal         Credit card  Electronic accessories    7.925000
Female  Normal         Cash         Food and beverages        7.876923
Male    Member         Credit card  Electronic accessories    7.843750
                       Ewallet      Health and beauty         7.815385
Female  Normal         Cash         Health and beauty         7.738462
Male    Normal         Ewallet      Sports and travel         7.584211
Female  Member         Ewallet      Home and lifestyle        7.520000
                       Cash         Food and beverages        7.485000
        Normal         Cash         Fashion accessories       7.476471
Name: Rating, dtype: float64
```

Out[57]: <Axes: title={'center': 'Top Groups With The Highest Average Customer Ratings'}, xlabel='Gender,Customer type,Payment,Product line'>



19. What are the characteristics of the most dissatisfied customers in terms of gender, customer type, payment method, and product line?

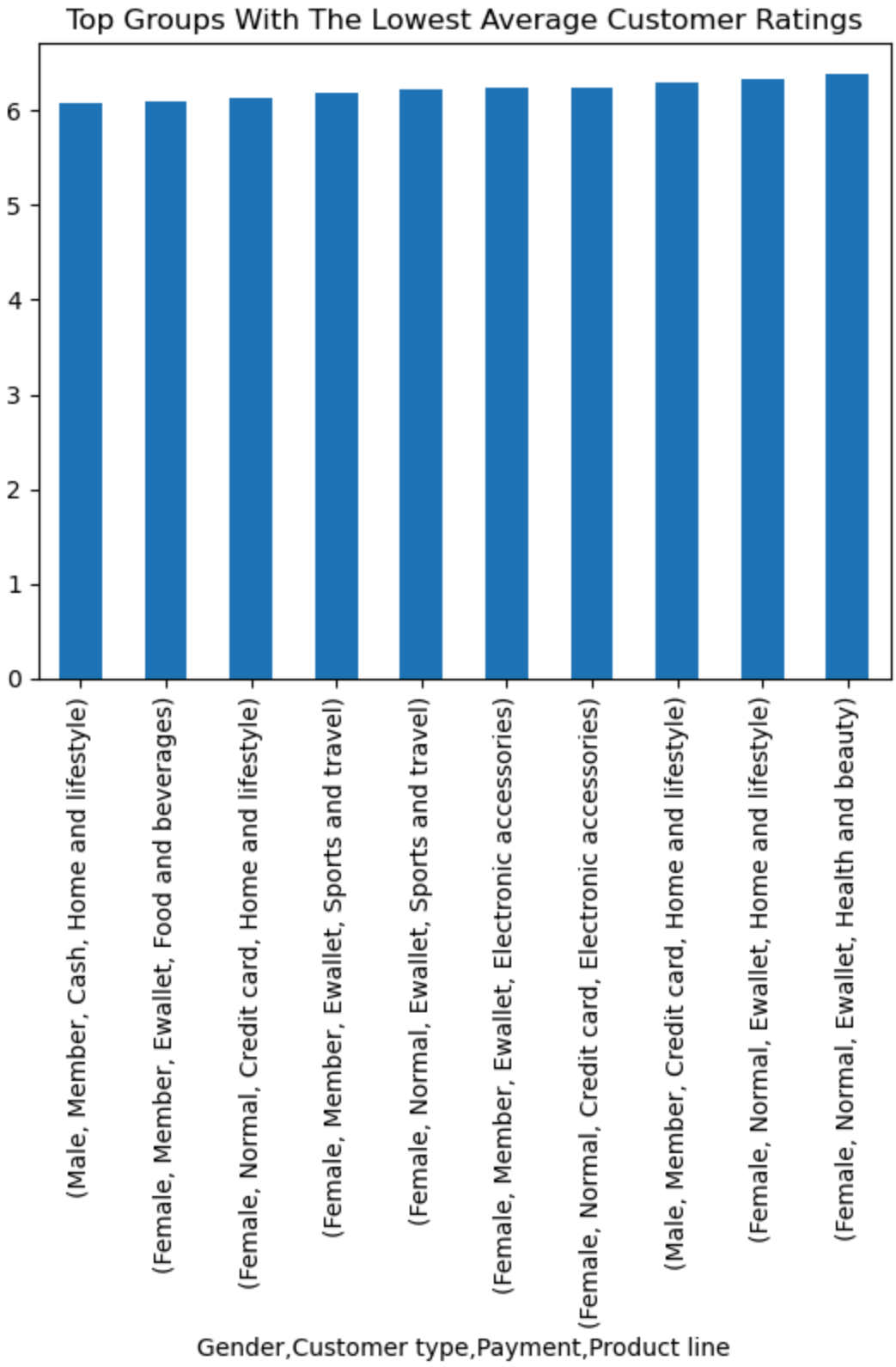
In [58]:
```
#group data by gender, customer type, payment method, and product line and calculate t
customer_dataa = df.groupby(['Gender', 'Customer type', 'Payment', 'Product line'])['R
```

```
#sort groups by average customer rating in ascending order
customer_dataa = customer_data.sort_values()

print('Top 10 groups with the lowest average customer ratings:')
print(customer_dataa.head(10))

customer_dataa.head(10).plot(kind='bar', title = 'Top Groups With The Lowest Average C
```

```
Top 10 groups with the lowest average customer ratings:
Gender  Customer type  Payment      Product line
Male    Member         Cash         Home and lifestyle        6.073333
Female  Member         Ewallet      Food and beverages        6.100000
        Normal         Credit card  Home and lifestyle        6.133333
        Member         Ewallet      Sports and travel         6.181250
        Normal         Ewallet      Sports and travel         6.227273
        Member         Ewallet      Electronic accessories    6.250000
        Normal         Credit card  Electronic accessories    6.250000
Male    Member         Credit card  Home and lifestyle        6.300000
Female  Normal         Ewallet      Home and lifestyle        6.335714
                                    Health and beauty         6.392857
Name: Rating, dtype: float64
```

Out[58]:

```
<Axes: title={'center': 'Top Groups With The Lowest Average Customer Ratings'}, xlabe
l='Gender,Customer type,Payment,Product line'>
```



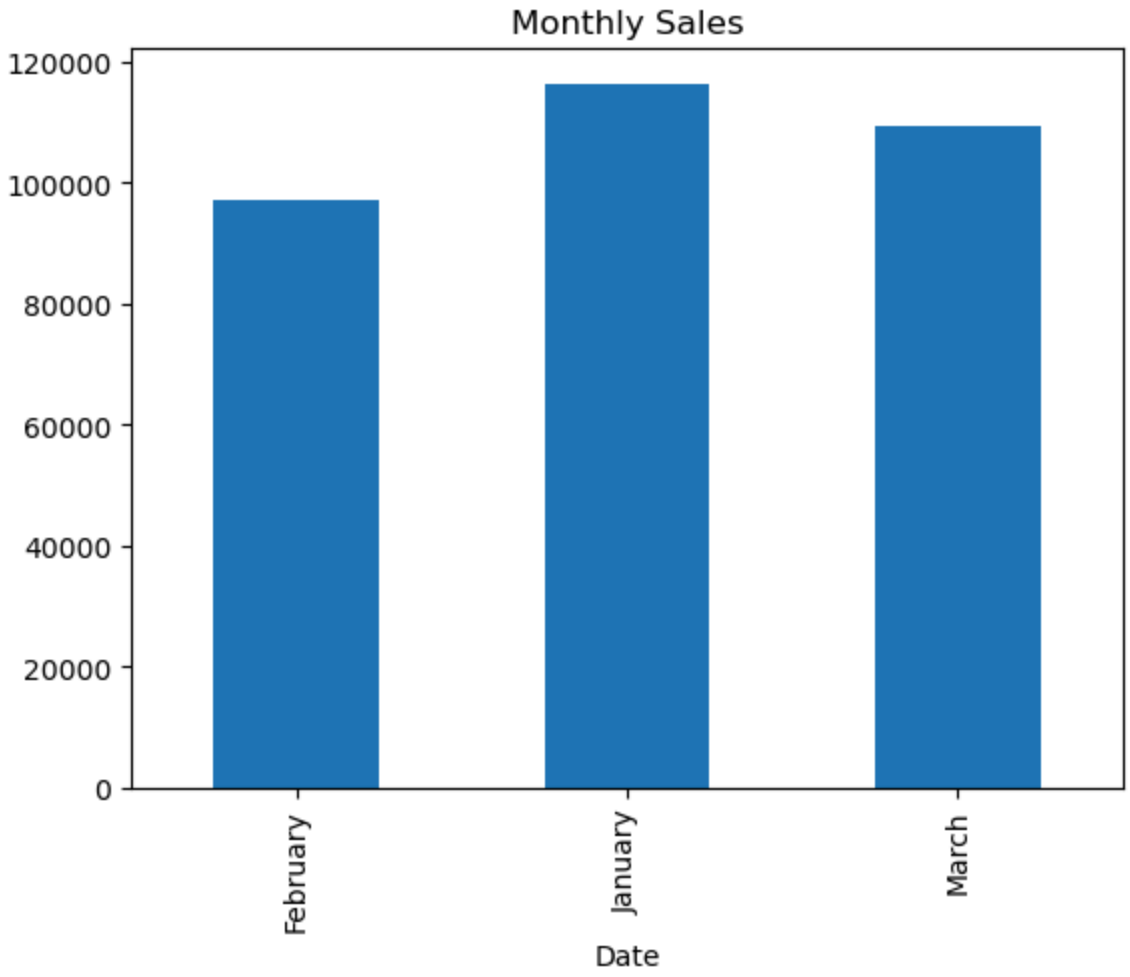
20. Analyze and visualize total sales per month, transactions per week, transactions per day, and transactions per hour

In [59]:
```
#calculate the total sales per month
df['Date'] = pd.to_datetime(df['Date'])
monthly_sales = df.groupby(df['Date'].dt.strftime('%B'))['Total'].sum()

print(monthly_sales)
```

```
monthly_sales.plot(kind='bar', title = 'Monthly Sales')
```

```
Date
February     97219.374
January     116291.868
March       109455.507
Name: Total, dtype: float64
```

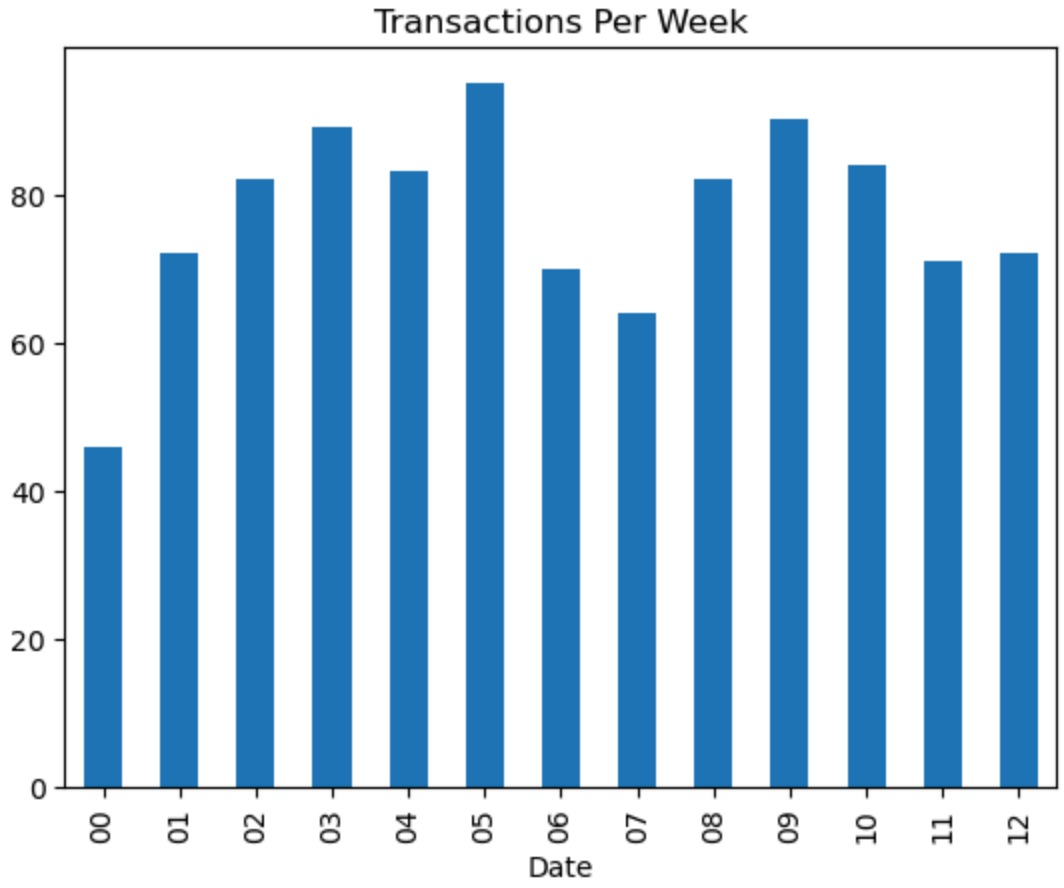
Out[59]: <Axes: title={'center': 'Monthly Sales'}, xlabel='Date'>



In [60]:
```
#calculate total number of transactions per week
df['Date'] = pd.to_datetime(df['Date'])
transactions_per_week = df.groupby(df['Date'].dt.strftime('%U'))['Invoice ID'].nunique

print(transactions_per_week)

transactions_per_week.plot(kind='bar', title = 'Transactions Per Week')
```

```
Date
00    46
01    72
02    82
03    89
04    83
05    95
06    70
07    64
08    82
09    90
10    84
11    71
12    72
Name: Invoice ID, dtype: int64
```

Out[60]: <Axes: title={'center': 'Transactions Per Week'}, xlabel='Date'>

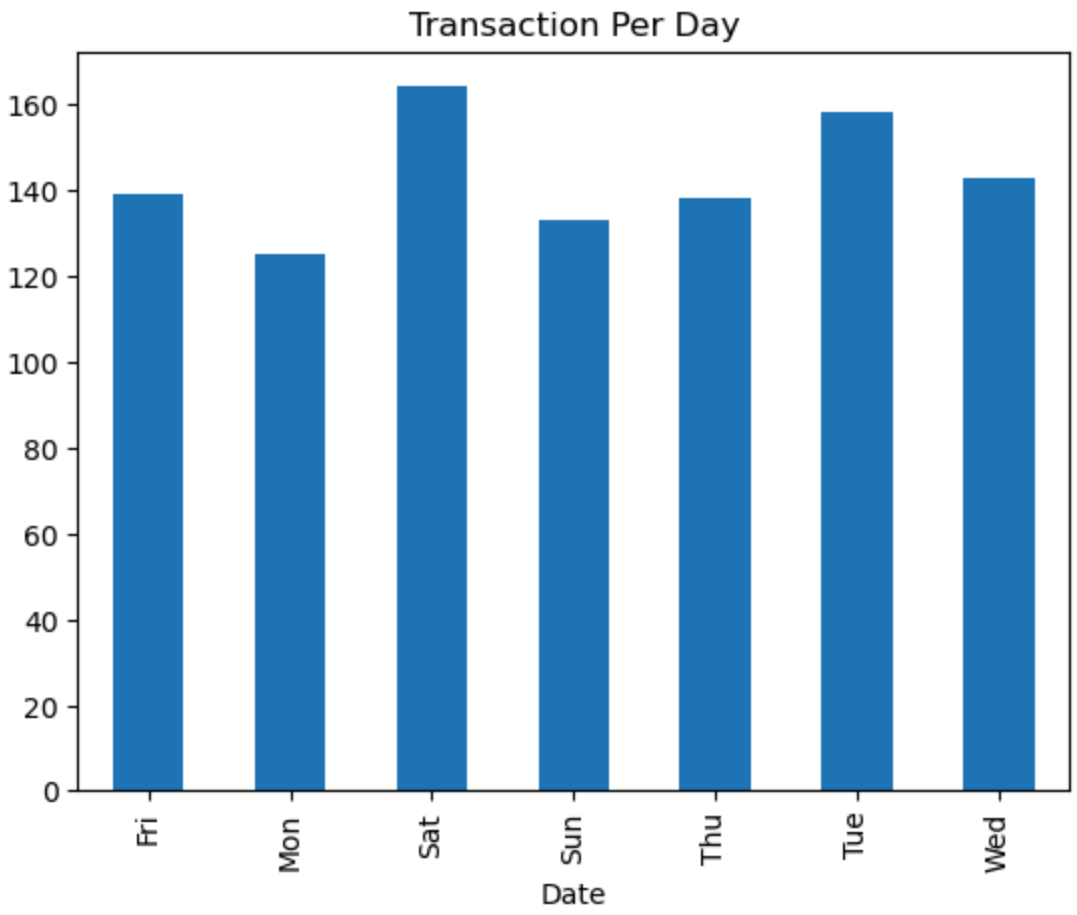


In [61]:
```
#calculate  total number of transactions per day
df['Date'] = pd.to_datetime(df['Date'])
transactions_per_day = df.groupby(df['Date'].dt.strftime('%a'))['Invoice ID'].nunique(

print(transactions_per_day)

transactions_per_day.plot(kind='bar', title='Transaction Per Day')
```

```
Date
Fri    139
Mon    125
Sat    164
Sun    133
Thu    138
Tue    158
Wed    143
Name: Invoice ID, dtype: int64
```

Out[61]:

```
<Axes: title={'center': 'Transaction Per Day'}, xlabel='Date'>
```



In [63]:

```
#calculate  total number of transactions per hour
df['Time'] = pd.to_datetime(df['Time'])
transactions_per_hour = df.groupby(df['Time'].dt.strftime('%H'))['Invoice ID'].nunique

print(transactions_per_hour)
```

```
Time
10    101
11     90
12     89
13    103
14     83
15    102
16     77
17     74
18     93
19    113
20     75
Name: Invoice ID, dtype: int64
```

21. What are the most commonly purchased products on weekends, and how does this compare to weekdays?

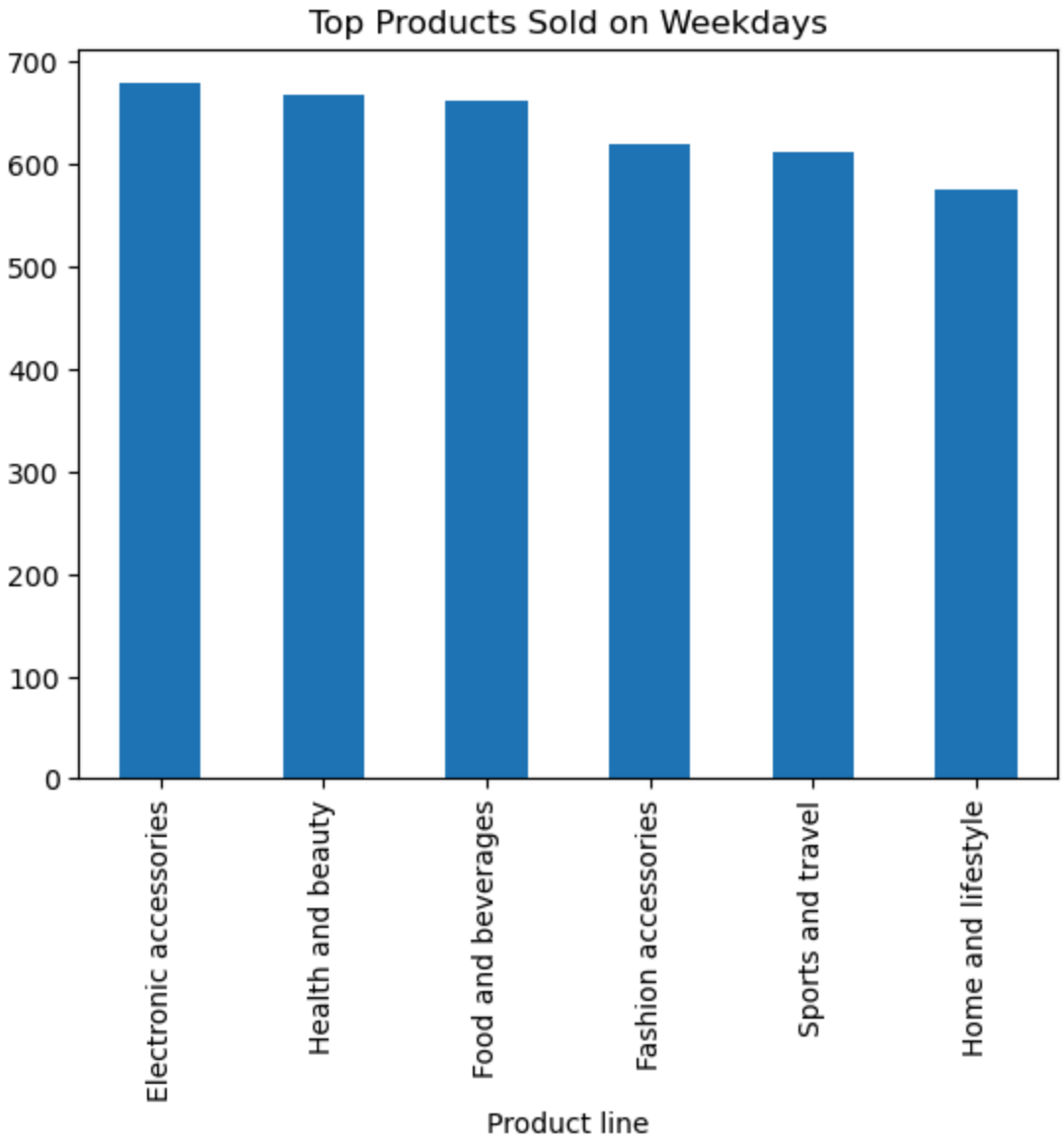
In [65]:
```
#group data by product line and day of the week and calculate the total quantity sold
product_data = df.groupby(['Product line', df['Date'].dt.strftime('%a')])['Quantity'].

weekend_data = product_data.loc[:, ['Sat', 'Sun']]
weekday_data = product_data.loc[:, ['Mon', 'Tue', 'Wed', 'Thu', 'Fri']]

weekend_data = weekend_data.groupby('Product line').sum()
weekday_data = weekday_data.groupby('Product line').sum()

weekend_data = weekend_data.sort_values(ascending=False)
weekday_data = weekday_data.sort_values(ascending=False)
```

Top products sold on weekdays

In [67]:
```
print('Top products sold on weekdays:')
print(weekday_data.head(10))

weekday_data.head(10).plot(kind='bar', title = 'Top Products Sold on Weekdays')
```

```
Top products sold on weekdays:
Product line
Electronic accessories    678
Health and beauty         667
Food and beverages        662
Fashion accessories       620
Sports and travel         611
Home and lifestyle        575
Name: Quantity, dtype: int64
```

Out[67]:
```
<Axes: title={'center': 'Top Products Sold on Weekdays'}, xlabel='Product line'>
```

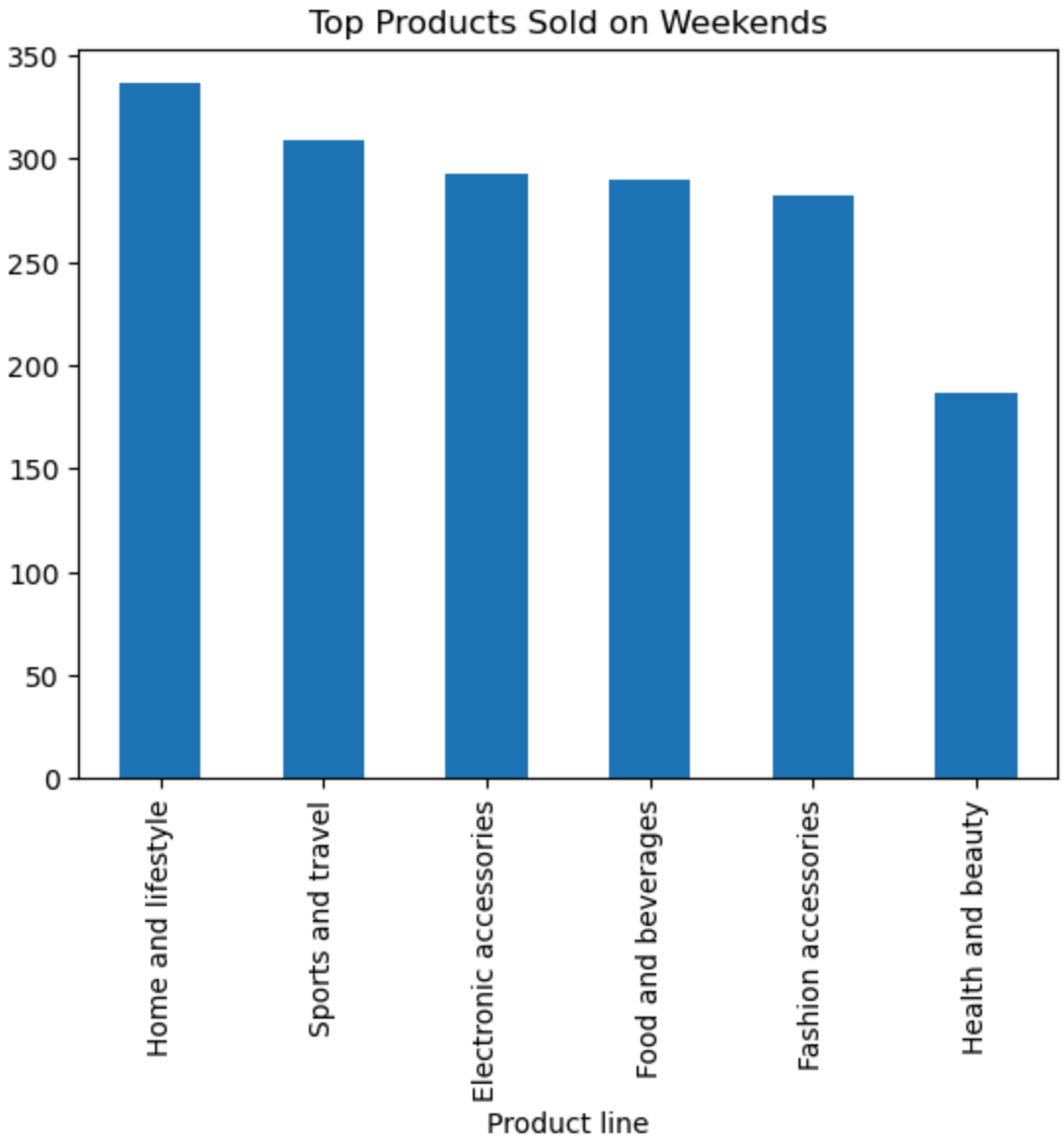


Top products sold on weekends

In [68]:
```
print('Top products sold on weekends:')
print(weekend_data.head(10))

weekend_data.head(10).plot(kind='bar', title = 'Top Products Sold on Weekends')
```

```
Top products sold on weekends:
Product line
Home and lifestyle        336
Sports and travel         309
Electronic accessories    293
Food and beverages        290
Fashion accessories       282
Health and beauty         187
Name: Quantity, dtype: int64
```

Out[68]:
```
<Axes: title={'center': 'Top Products Sold on Weekends'}, xlabel='Product line'>
```



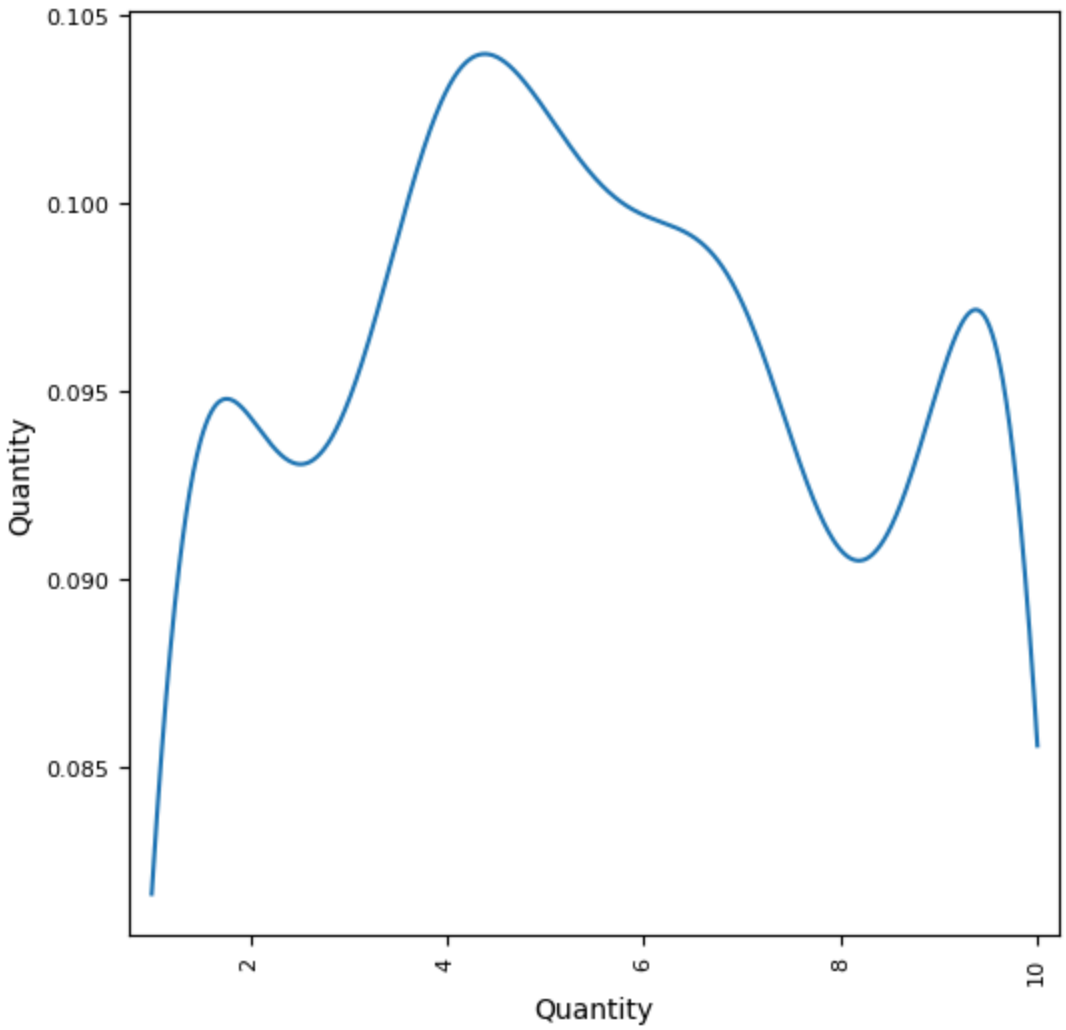
22. What is the relationship between the date and time of a transaction and the customer's gender, type, and payment method?

In [69]:
```
#group  data by gender, customer type, payment method, and date and time and calculate
transaction_data = df.groupby(['Gender', 'Customer type', 'Payment', 'Date', 'Time'])[

print('Top 10 groups with the highest total quantity sold:')
print(transaction_data.sort_values(ascending=False).head(10))
```

```
Top 10 groups with the highest total quantity sold:
Gender  Customer type  Payment      Date        Time
Male    Normal         Ewallet      2019-03-29  2024-04-18 10:25:00    10
        Member         Credit card  2019-02-14  2024-04-18 11:26:00    10
Female  Member         Ewallet      2019-01-30  2024-04-18 20:23:00    10
                                    2019-02-10  2024-04-18 12:28:00    10
                                    2019-02-22  2024-04-18 12:30:00    10
Male    Member         Credit card  2019-03-20  2024-04-18 19:57:00    10
Female  Member         Ewallet      2019-03-05  2024-04-18 16:24:00    10
                                    2019-03-08  2024-04-18 10:53:00    10
Male    Member         Credit card  2019-03-17  2024-04-18 19:06:00    10
Female  Member         Ewallet      2019-03-18  2024-04-18 17:38:00    10
Name: Quantity, dtype: int64
```

In [70]:

```python
from pandas.plotting import scatter_matrix

transaction_data = transaction_data.to_frame()

#create scatter matrix plot of the data
scatter_matrix(transaction_data, alpha=0.2, figsize=(6, 6), diagonal='kde')

plt.show()
```



CONCLUSION

Based on the analysis of the supermarket dataset, it is recommended to focus on:

- Increasing the average unit price and gross margin percentage can help maximize profits.

- Promoting product lines with high sales quantities can also boost revenue.

- Targeting specific cities and customer types can lead to more transactions.

- Improving the supermarket's rating across genders can enhance customer satisfaction and loyalty.

- Implementing strategies to minimize tax payments can further improve overall profitability.

- Incorporating additional dataset attributes can further improve the accuracy of the sales predictions, which can inform business decisions and resource allocation.