Import needed Libraries

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Reading the Data

In [2]:

```
df = pd.read_csv("/kaggle/input/customer-shopping-
dataset/customer_shopping_data.csv")
```

Attribute Information:
* invoice_no: Invoice number. Nominal. A combination of the letter 'I' and a 6-digit integer uniquely assigned to each operation.
* customer_id: Customer number. Nominal. A combination of the letter 'C' and a 6-digit integer uniquely assigned to each operation.
* gender: String variable of the customer's gender.
* age: Positive Integer variable of the customers age.
* category: String variable of the category of the purchased product.
* quantity: The quantities of each product (item) per transaction. Numeric.
* price: Unit price. Numeric. Product price per unit in Turkish Liras (TL).
* payment_method: String variable of the payment method (cash, credit card or debit card) used for the transaction.
* invoice_date: Invoice date. The day when a transaction was generated.
* shopping_mall: String variable of the name of the shopping mall where the transaction was made

Exploring the Data

In [3]:

```
df.shape
```

Out[3]:

```
(99457, 10)
```

In [4]:

```
df.head()
```

Out[4]:

	invoice_no	customer_id	gender	age	category	quantity	pric
0	I138884	C241288	Female	28	Clothing	5	1500.4
1	I317333	C111565	Male	21	Shoes	3	1800.5
2	I127801	C266599	Male	20	Clothing	1	300.08
3	I173702	C988172	Female	66	Shoes	5	3000.8
4	I337046	C189076	Female	53	Books	4	60.60

In [5]:

```
df.tail()
```

Out[5]:

	invoice_no	customer_id	gender	age	category	quantity
99452	I219422	C441542	Female	45	Souvenir	5
99453	I325143	C569580	Male	27	Food & Beverage	2
99454	I824010	C103292	Male	63	Food & Beverage	2
99455	I702964	C800631	Male	56	Technology	4
99456	I232867	C273973	Female	36	Souvenir	3

In [6]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99457 entries, 0 to 99456
Data columns (total 10 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   invoice_no      99457 non-null  object 
 1   customer_id     99457 non-null  object 
 2   gender          99457 non-null  object 
 3   age             99457 non-null  int64  
 4   category        99457 non-null  object 
 5   quantity        99457 non-null  int64  
 6   price           99457 non-null  float64
 7   payment_method  99457 non-null  object 
 8   invoice_date    99457 non-null  object 
 9   shopping_mall   99457 non-null  object 
dtypes: float64(1), int64(2), object(7)
memory usage: 7.6+ MB
```

when make discribe to data as null or not we found not null but found wrong in the data type in column invoice date should be datetime and anthor column but we not want to change the types for this columns

In [7]:

```
df.describe()
```

Out[7]:

	age	quantity	price
count	99457.000000	99457.000000	99457.000000
mean	43.427089	3.003429	689.256321
std	14.990054	1.413025	941.184567
min	18.000000	1.000000	5.230000
25%	30.000000	2.000000	45.450000
50%	43.000000	3.000000	203.300000
75%	56.000000	4.000000	1200.320000
max	69.000000	5.000000	5250.000000

From describe the stat info we found outlier in price column when mean is 689.256321 and the min 5.230 and the max is 5250 then there are outliers in this column

In [8]:
```
df.isna().sum()
```

Out[8]:
```
invoice_no        0
customer_id       0
gender            0
age               0
category          0
quantity          0
price             0
payment_method    0
invoice_date      0
shopping_mall     0
dtype: int64
```

No have nan value in the data

In [9]:
```
df.duplicated().sum()
```

Out[9]:
```
0
```

No have Duplicated value in the data

Cleaning the Data

In [10]:
```
df.columns
```

Out[10]:
```
Index(['invoice_no', 'customer_id', 'gender', 'age', 'category',
'quantity',
       'price', 'payment_method', 'invoice_date', 'shopping_mall'],
      dtype='object')
```

In [11]:
```
df['invoice_date'] = pd.to_datetime(df['invoice_date'], format='%d/%m/%Y')
```

Change type of the invoice_date column to can deal with this column

In [12]:
```
df['year'] = df['invoice_date'].dt.year
df['month'] = df['invoice_date'].dt.month
```

In [13]:

```
df.head()
```

Out[13]:

	invoice_no	customer_id	gender	age	category	quantity	pric
0	I138884	C241288	Female	28	Clothing	5	1500.4
1	I317333	C111565	Male	21	Shoes	3	1800.5
2	I127801	C266599	Male	20	Clothing	1	300.08
3	I173702	C988172	Female	66	Shoes	5	3000.8
4	I337046	C189076	Female	53	Books	4	60.60

Add 2 colmns to make easier to deal with years and months

In [14]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99457 entries, 0 to 99456
Data columns (total 12 columns):
 #   Column          Non-Null Count  Dtype         
---  ------          --------------  -----         
 0   invoice_no      99457 non-null  object        
 1   customer_id     99457 non-null  object        
 2   gender          99457 non-null  object        
 3   age             99457 non-null  int64         
 4   category        99457 non-null  object        
 5   quantity        99457 non-null  int64         
 6   price           99457 non-null  float64       
 7   payment_method  99457 non-null  object        
 8   invoice_date    99457 non-null  datetime64[ns]
 9   shopping_mall   99457 non-null  object        
 10  year            99457 non-null  int32         
 11  month           99457 non-null  int32         
dtypes: datetime64[ns](1), float64(1), int32(2), int64(2), object(6)
memory usage: 8.3+ MB
```

we can see the change that happen in type of the date column and add two another columns to data

Analysis and Visualization

In [15]:
```
df.head()
```

Out[15]:

	invoice_no	customer_id	gender	age	category	quantity	pric
0	I138884	C241288	Female	28	Clothing	5	1500.4
1	I317333	C111565	Male	21	Shoes	3	1800.5
2	I127801	C266599	Male	20	Clothing	1	300.08
3	I173702	C988172	Female	66	Shoes	5	3000.8
4	I337046	C189076	Female	53	Books	4	60.60

Gender

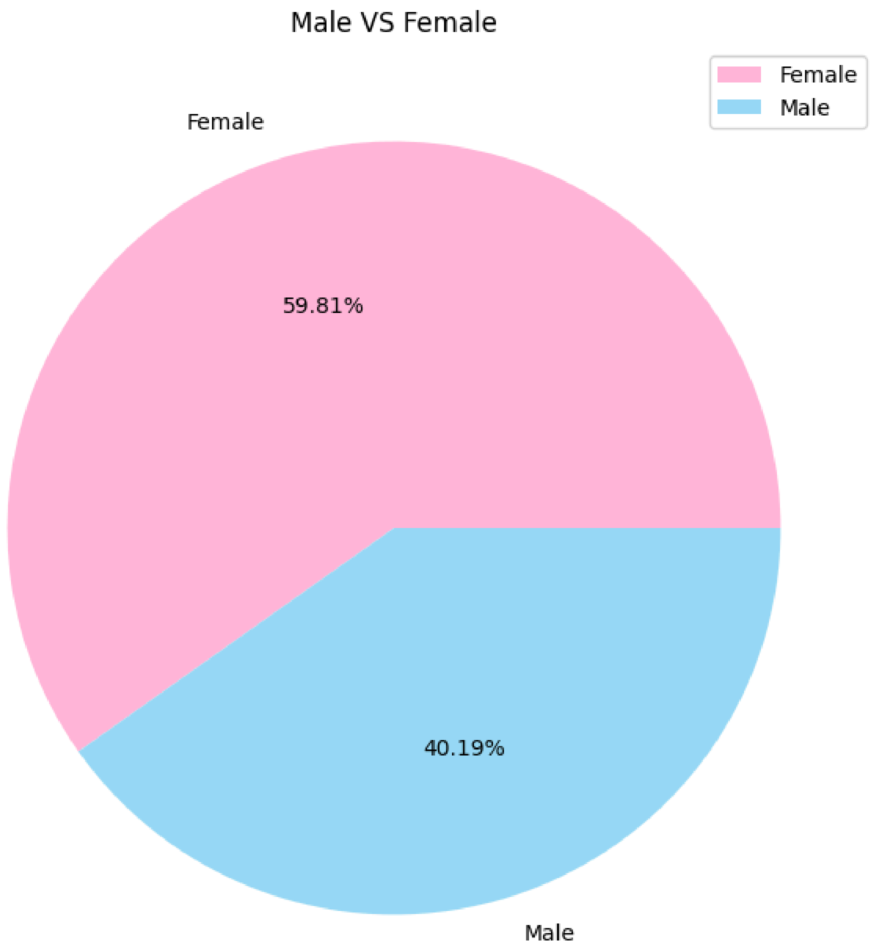
In [16]:
```
Gender = df['gender'].value_counts().reset_index()
Gender
```

Out[16]:

	gender	count
0	Female	59482
1	Male	39975

In [17]:

```
plt.figure(figsize=(8,8))
plt.pie(Gender['count'],
        labels=Gender['gender'],autopct='%1.2f%%',
        colors=["#FFB6D9", "#99DBF5"]
       )
plt.title('Male VS Female')
plt.legend()
plt.show()
```



From this pie graph we can see the female is more than male when do shopping

Gender with Category

In [18]:

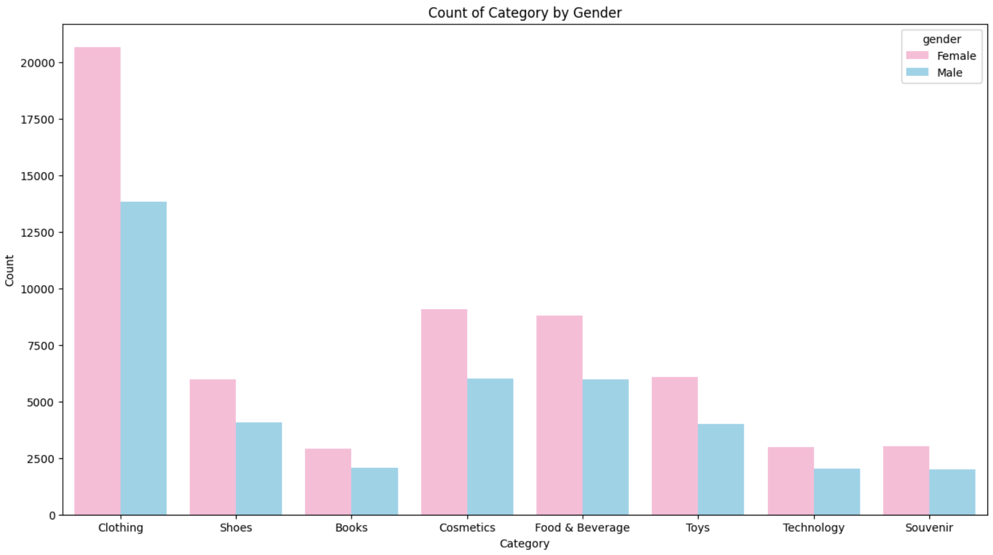
```
Genderwithcatergory = df.groupby('gender')
['category'].value_counts().reset_index()
Genderwithcatergory
```

Out[18]:

	gender	category	count
0	Female	Clothing	20652
1	Female	Cosmetics	9070
2	Female	Food & Beverage	8804
3	Female	Toys	6085
4	Female	Shoes	5967
5	Female	Souvenir	3017
6	Female	Technology	2981
7	Female	Books	2906
8	Male	Clothing	13835
9	Male	Cosmetics	6027
10	Male	Food & Beverage	5972
11	Male	Shoes	4067
12	Male	Toys	4002
13	Male	Books	2075
14	Male	Technology	2015
15	Male	Souvenir	1982

In [19]:

```
plt.figure(figsize=(15, 8))
sns.countplot(data=df, x=df['category'], hue=df['gender'],palette=
["#FFB6D9", "#99DBF5"])
plt.title('Count of Category by Gender')
plt.xlabel('Category')
plt.ylabel('Count')
plt.show()
```



From this graph we can see the most type of male and female buy colthing and few but books and tech and souvenir

Gender with average age

In [20]:

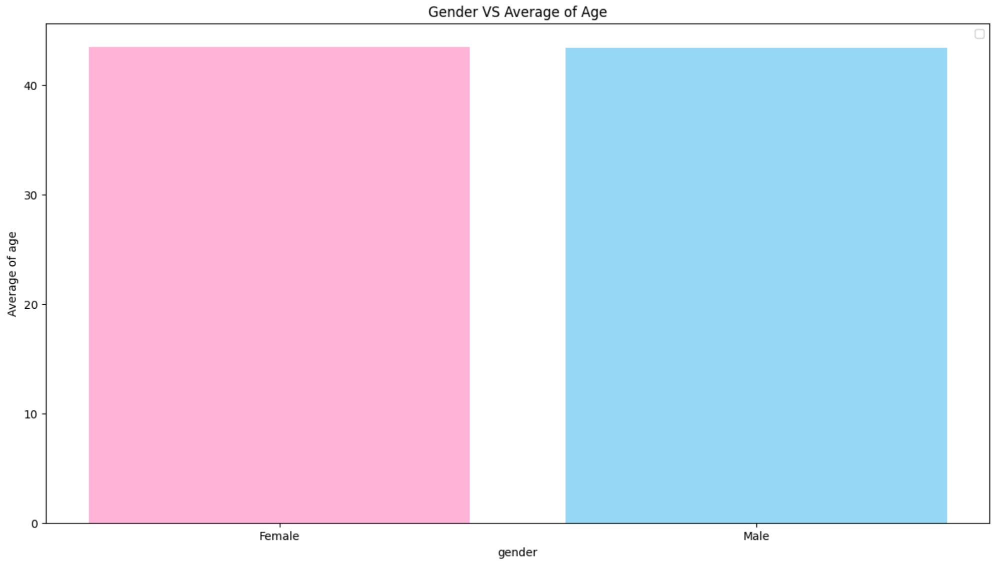
```
Genderwithavgage = df.groupby('gender')['age'].mean().reset_index()
Genderwithavgage
```

Out[20]:

	gender	age
0	Female	43.453515
1	Male	43.387767

In [21]:

```
plt.figure(figsize=(15, 8))
colors = ["#FFB6D9", "#99DBF5"]
plt.bar(Genderwithavgage['gender'],Genderwithavgage['age'], color=colors)
plt.xlabel('gender')
plt.ylabel('Average of age')
plt.title('Gender VS Average of Age')
plt.legend()
plt.show()
```



This Graph show the Average of male and female in the data

Gender with Price

In [22]:

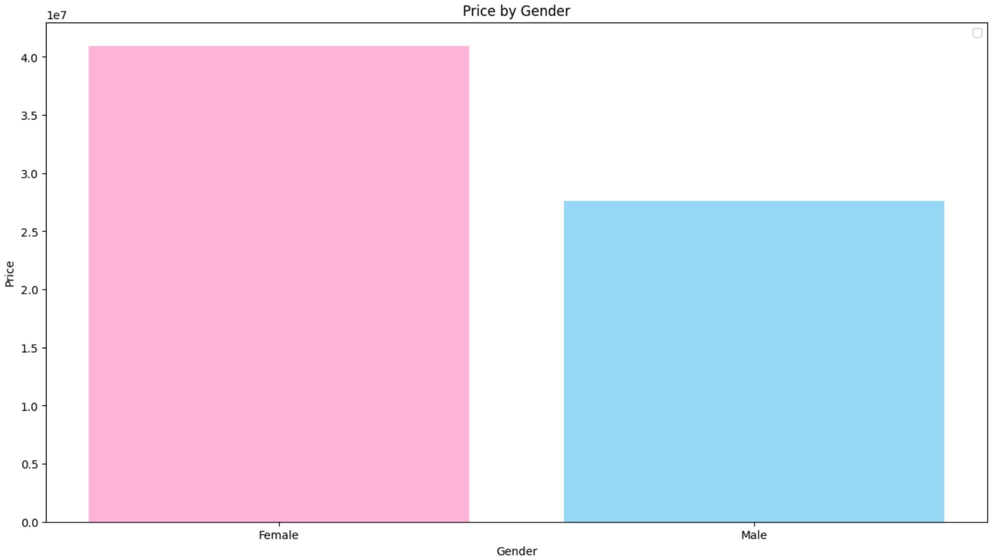
```
Genderwithprice = df.groupby('gender')['price'].sum().reset_index()
Genderwithprice
```

Out[22]:

	gender	price
0	Female	40931801.62
1	Male	27619564.29

In [23]:

```
plt.figure(figsize=(15, 8))
colors = ["#FFB6D9", "#99DBF5"]
plt.bar(Genderwithprice['gender'],Genderwithprice['price'], color=colors)
plt.xlabel('Gender')
plt.ylabel('Price')
plt.title('Price by Gender')
plt.legend()
plt.show()
```



from this Graph we can see the female pay money more than male

In [24]:

```
df.head()
```

Out[24]:

	invoice_no	customer_id	gender	age	category	quantity	pric
0	I138884	C241288	Female	28	Clothing	5	1500.4
1	I317333	C111565	Male	21	Shoes	3	1800.5
2	I127801	C266599	Male	20	Clothing	1	300.08
3	I173702	C988172	Female	66	Shoes	5	3000.8
4	I337046	C189076	Female	53	Books	4	60.60

Age

In [25]:

```
age = df['age'].value_counts().reset_index()
age
```

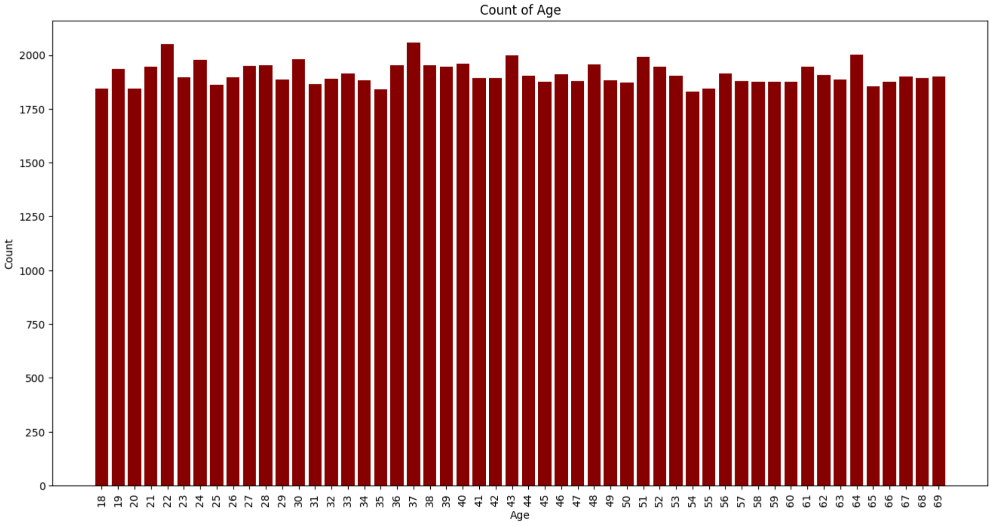
Out[25]:

	age	count
0	37	2057
1	22	2051
2	64	2002
3	43	2000
4	51	1993
5	30	1981
6	24	1977
7	40	1960
8	48	1955
9	36	1954
10	38	1954
11	28	1953
12	27	1950
13	39	1947
14	21	1947
15	61	1945
16	52	1945
17	19	1936
18	56	1916
19	33	1913
20	46	1911
21	62	1909
22	44	1904

	age	count
23	53	1903
24	67	1901
25	69	1901
26	23	1897
27	26	1896
28	68	1893
29	42	1892
30	41	1892
31	32	1891
32	63	1886
33	29	1885
34	49	1883
35	34	1883
36	47	1880
37	57	1879
38	66	1876
39	45	1876
40	58	1875
41	59	1874
42	60	1874
43	50	1873
44	31	1866
45	25	1863
46	65	1856
47	18	1844
48	20	1844
49	55	1843
50	35	1841
51	54	1830

In [26]:

```
xticks = age['age']
plt.figure(figsize=(16, 8))
plt.bar(age['age'], age['count'],color ='darkred')
plt.xticks(xticks,rotation=90)
plt.xlabel('Age')
plt.ylabel('Count')
plt.title('Count of Age')
plt.show()
```



From this graph we can determine the number for every age in data

Age with Quantity

In [27]:

```
AgewithQuantity = df.groupby('age')['quantity'].sum().reset_index()
AgewithQuantity
```

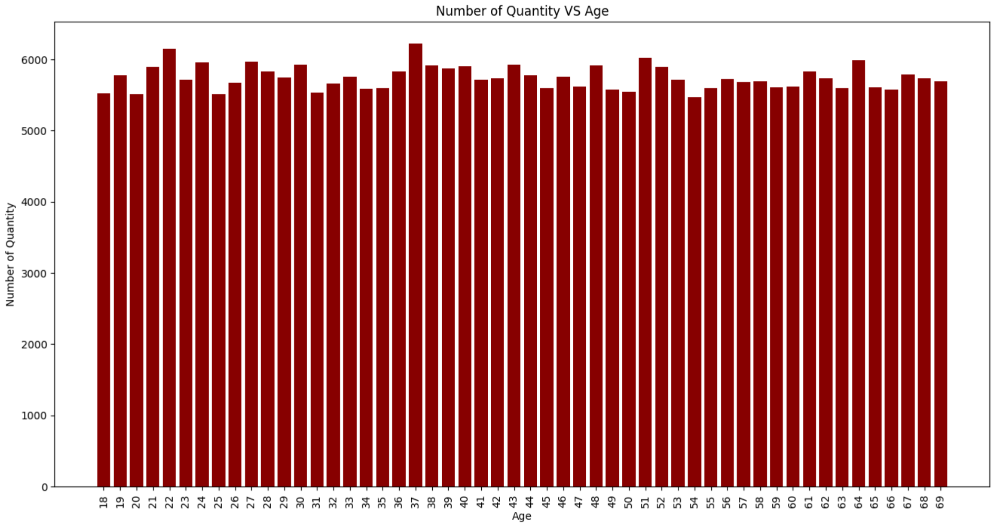
Out[27]:

	age	quantity
0	18	5518
1	19	5778
2	20	5505
3	21	5894
4	22	6148
5	23	5715
6	24	5957
7	25	5513
8	26	5672
9	27	5969
10	28	5832
11	29	5744
12	30	5927
13	31	5531
14	32	5655
15	33	5756
16	34	5589
17	35	5590
18	36	5826
19	37	6217
20	38	5910
21	39	5874
22	40	5902

	age	quantity
23	41	5715
24	42	5734
25	43	5928
26	44	5777
27	45	5600
28	46	5751
29	47	5617
30	48	5918
31	49	5575
32	50	5541
33	51	6014
34	52	5892
35	53	5712
36	54	5471
37	55	5595
38	56	5727
39	57	5675
40	58	5692
41	59	5603
42	60	5616
43	61	5829
44	62	5734
45	63	5591
46	64	5991
47	65	5607
48	66	5571
49	67	5788
50	68	5737
51	69	5689

In [28]:

```
xticks = AgewithQuantity['age']
plt.figure(figsize=(16, 8))
plt.bar(AgewithQuantity['age'], AgewithQuantity['quantity'],color
="darkred")
plt.xticks(xticks,rotation=90)
plt.xlabel('Age')
plt.ylabel('Number of Quantity')
plt.title('Number of Quantity VS Age')
plt.show()
```



this Graph show the number of quantity for every age

Category with Quantity

In [29]:

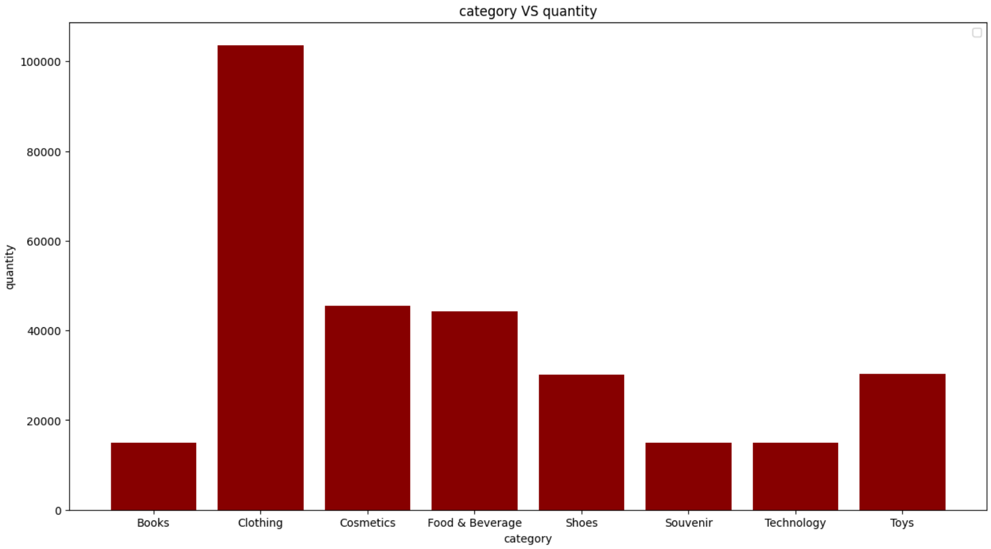
```
Categorywithquantity = df.groupby('category')
['quantity'].sum().reset_index()
Categorywithquantity
```

Out[29]:

	category	quantity
0	Books	14982
1	Clothing	103558
2	Cosmetics	45465
3	Food & Beverage	44277
4	Shoes	30217
5	Souvenir	14871
6	Technology	15021
7	Toys	30321

In [30]:

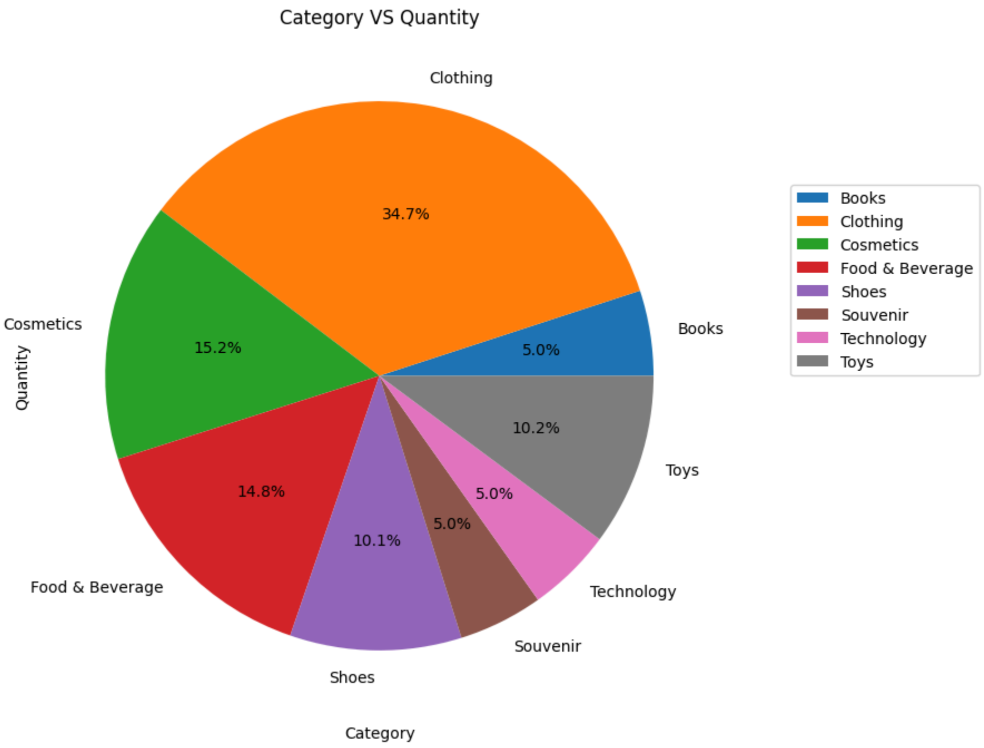
```
plt.figure(figsize=(15, 8))
plt.bar(Categorywithquantity['category'],Categorywithquantity['quantity'],c
olor ="darkred")
plt.xlabel('category')
plt.ylabel('quantity')
plt.title('category VS quantity')
plt.legend()
plt.show()
```



From this bar graph we can see the quantity for all Category that customer buy

In [31]:

```
plt.figure(figsize=(8, 8))
plt.pie(Categorywithquantity['quantity'],
labels=Categorywithquantity['category'], autopct='%1.1f%%')
plt.xlabel('Category')
plt.ylabel('Quantity')
plt.title('Category VS Quantity')
plt.legend(loc=(1.1,.5))
plt.show()
```



From this pie graph we can see the percentage of quantity for all Category that customer buy

Category with Price

In [32]:

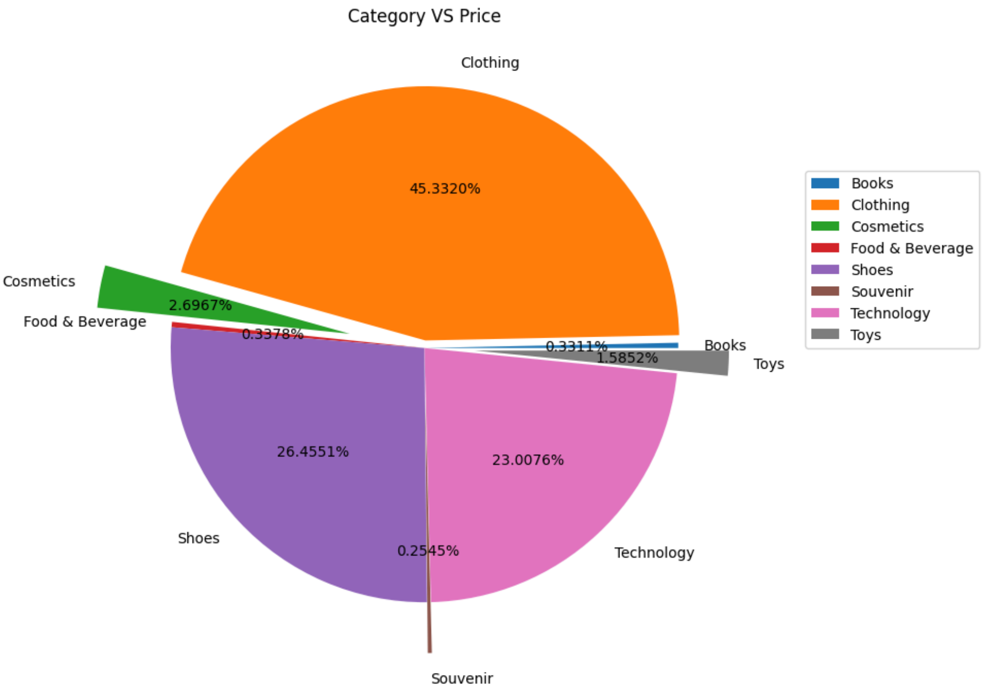
```
CategorywithPrice = df.groupby('category')['price'].sum().reset_index()
CategorywithPrice
```

Out[32]:

	category	price
0	Books	226977.30
1	Clothing	31075684.64
2	Cosmetics	1848606.90
3	Food & Beverage	231568.71
4	Shoes	18135336.89
5	Souvenir	174436.83
6	Technology	15772050.00
7	Toys	1086704.64

In [33]:

```
plt.figure(figsize=(8,8))
plt.pie(CategorywithPrice['price'],
        labels=CategorywithPrice['category'],autopct='%1.4f%%',
        explode=[0.0,0.03,0.3,0,0,0.2,0,0.2]
       )
plt.title('Category VS Price')
plt.legend(loc=(1.1,.5))
plt.show()
```



From this pie graph we can see the percentage of price for all Category

In [34]:

```
df.head()
```

Out[34]:

	invoice_no	customer_id	gender	age	category	quantity	pric
0	I138884	C241288	Female	28	Clothing	5	1500.4
1	I317333	C111565	Male	21	Shoes	3	1800.5
2	I127801	C266599	Male	20	Clothing	1	300.08
3	I173702	C988172	Female	66	Shoes	5	3000.8
4	I337046	C189076	Female	53	Books	4	60.60

Price with years

In [35]:

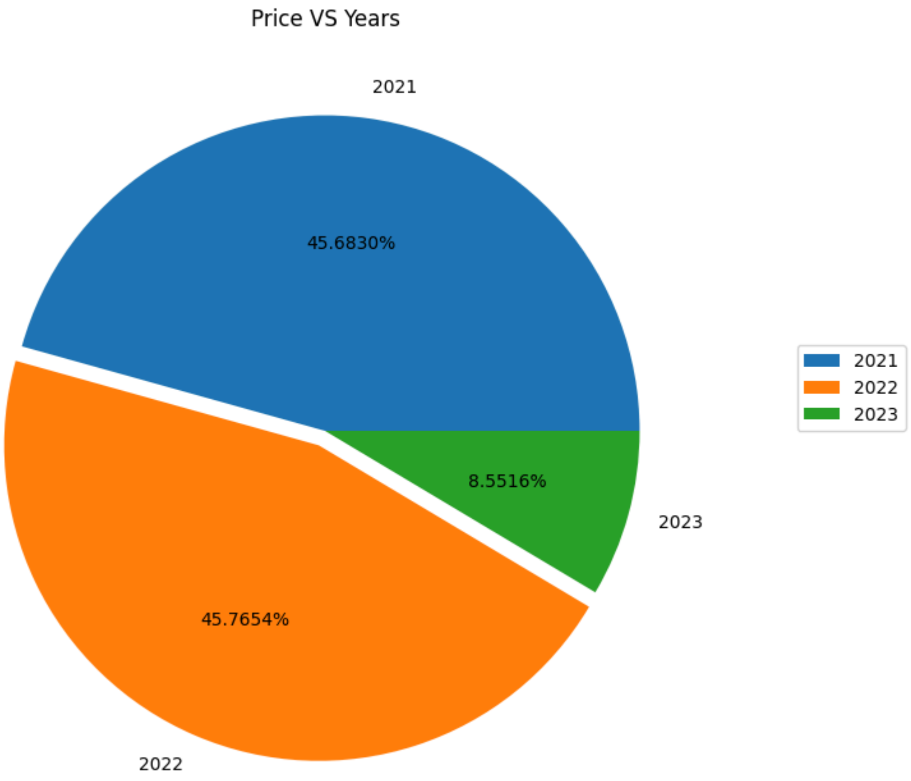
```
Pricewithyears = df.groupby('year')['price'].sum().reset_index()
Pricewithyears
```

Out[35]:

	year	price
0	2021	31316304.63
1	2022	31372826.18
2	2023	5862235.10

In [36]:

```
plt.figure(figsize=(8,8))
plt.pie(Pricewithyears['price'],
        labels=Pricewithyears['year'],autopct='%1.4f%%',
        explode=[0,0.05,0]
       )
plt.title('Price VS Years')
plt.legend(loc=(1.1,.5))
plt.show()
```



From this pie we can see the biggest money get from sell is in 2021 and 2022

payment_method

In [37]:

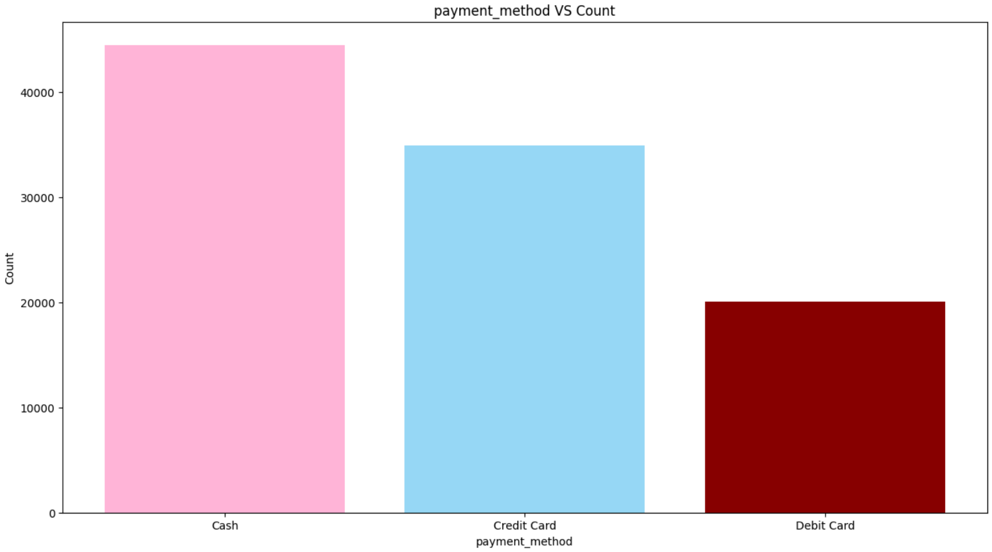
```
payment_method = df['payment_method'].value_counts().reset_index()
payment_method
```

Out[37]:

	payment_method	count
0	Cash	44447
1	Credit Card	34931
2	Debit Card	20079

In [38]:

```
plt.figure(figsize=(15, 8))
plt.bar(payment_method['payment_method'],payment_method['count'],color=
["#FFB6D9", "#99DBF5",'darkred'])
plt.title('payment_method VS Count')
plt.xlabel('payment_method')
plt.ylabel('Count')
plt.show()
```



This bar show the number of operation that happen when way and how the cutomer pay

In [39]:

```
df.head()
```

Out[39]:

	invoice_no	customer_id	gender	age	category	quantity	pric
0	I138884	C241288	Female	28	Clothing	5	1500.4
1	I317333	C111565	Male	21	Shoes	3	1800.5
2	I127801	C266599	Male	20	Clothing	1	300.08
3	I173702	C988172	Female	66	Shoes	5	3000.8
4	I337046	C189076	Female	53	Books	4	60.60

shopping_mall

In [40]:

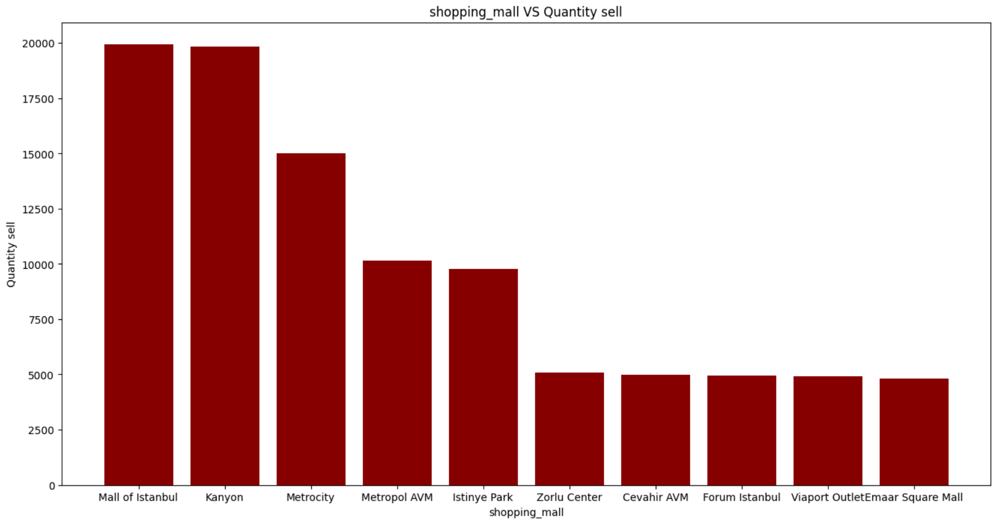
```
shopping_mall = df['shopping_mall'].value_counts().reset_index()
shopping_mall
```

Out[40]:

	shopping_mall	count
0	Mall of Istanbul	19943
1	Kanyon	19823
2	Metrocity	15011
3	Metropol AVM	10161
4	Istinye Park	9781
5	Zorlu Center	5075
6	Cevahir AVM	4991
7	Forum Istanbul	4947
8	Viaport Outlet	4914
9	Emaar Square Mall	4811

In [41]:

```
plt.figure(figsize=(16, 8))
plt.bar(shopping_mall['shopping_mall'],shopping_mall['count'],color='darkre
d')
plt.title('shopping_mall VS Quantity sell')
plt.xlabel('shopping_mall')
plt.ylabel('Quantity sell')
plt.show()
```



This graph show the number of Category that sell from all malls

Orders with Years

In [42]:

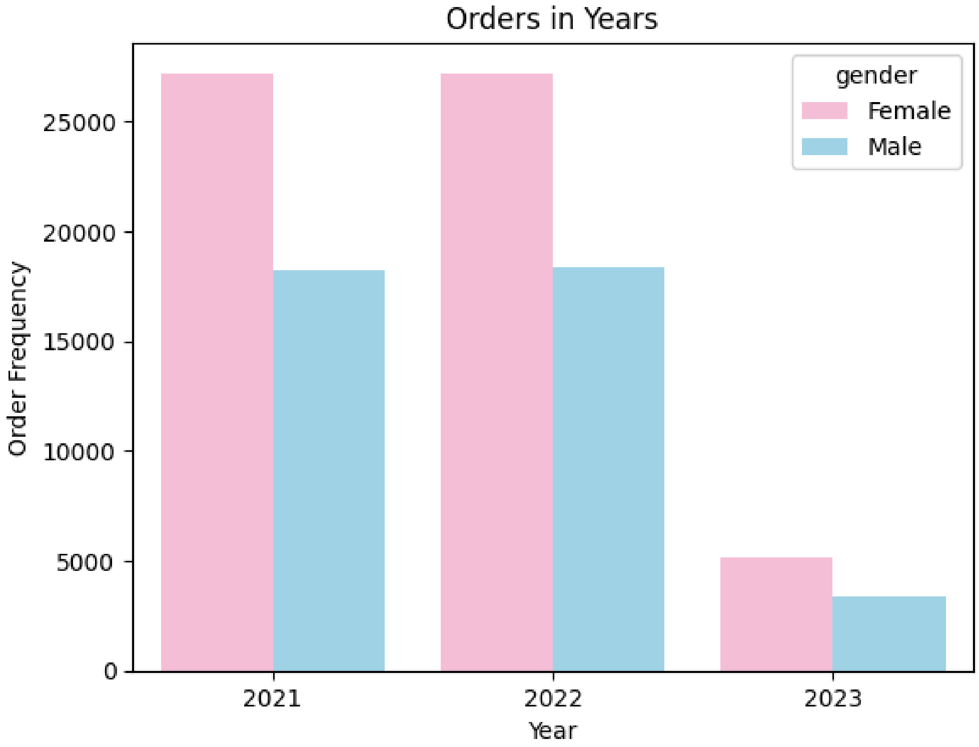
```
years_gender = df.groupby(['year', 'gender'])
['year'].value_counts().reset_index()
years_gender
```

Out[42]:

	year	gender	count
0	2021	Female	27156
1	2021	Male	18226
2	2022	Female	27192
3	2022	Male	18359
4	2023	Female	5134
5	2023	Male	3390

In [43]:

```
sns.barplot(data=years_gender, x='year', y='count', hue='gender',palette=
["#FFB6D9", "#99DBF5"])
plt.xlabel('Year')
plt.ylabel('Order Frequency')
plt.title('Orders in Years')
plt.show()
```



this Graph show the number of Orders in Years

Order in months

In [44]:

```
months_gender = df.groupby(['month', 'gender'])
['month'].value_counts().reset_index()
months_gender
```

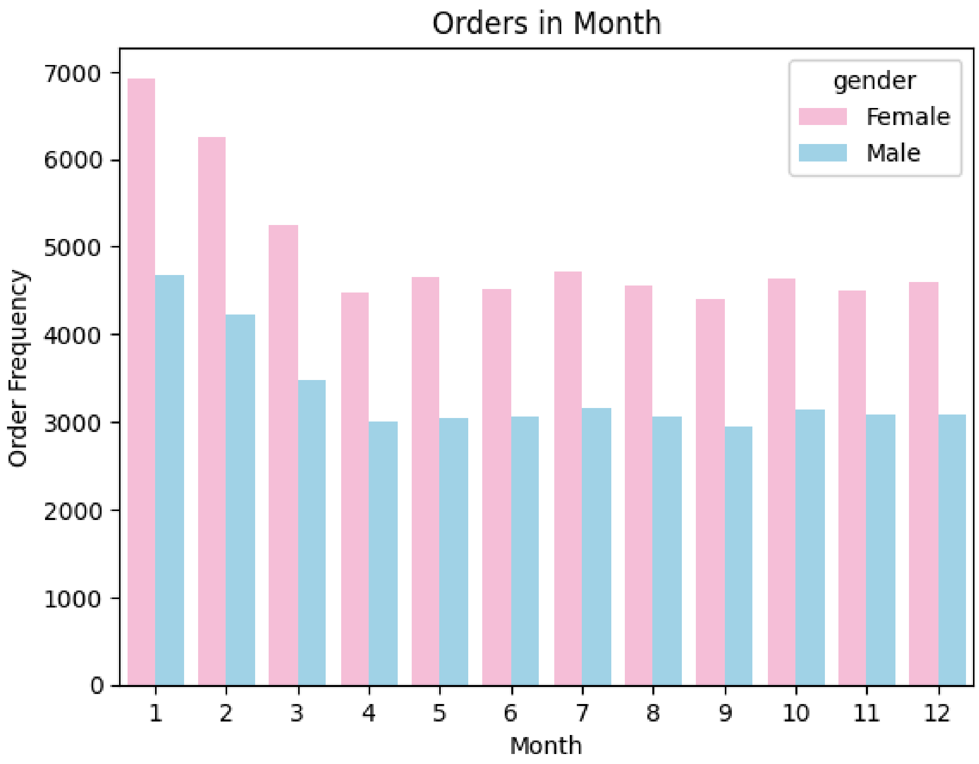
Out[44]:

	month	gender	count
0	1	Female	6923
1	1	Male	4685
2	2	Female	6252
3	2	Male	4230
4	3	Female	5248
5	3	Male	3482
6	4	Female	4481
7	4	Male	3006
8	5	Female	4649
9	5	Male	3048
10	6	Female	4518
11	6	Male	3063
12	7	Female	4723
13	7	Male	3154
14	8	Female	4567
15	8	Male	3068
16	9	Female	4404
17	9	Male	2949
18	10	Female	4632
19	10	Male	3132
20	11	Female	4489
21	11	Male	3074

	month	gender	count
22	12	Female	4596
23	12	Male	3084

In [45]:

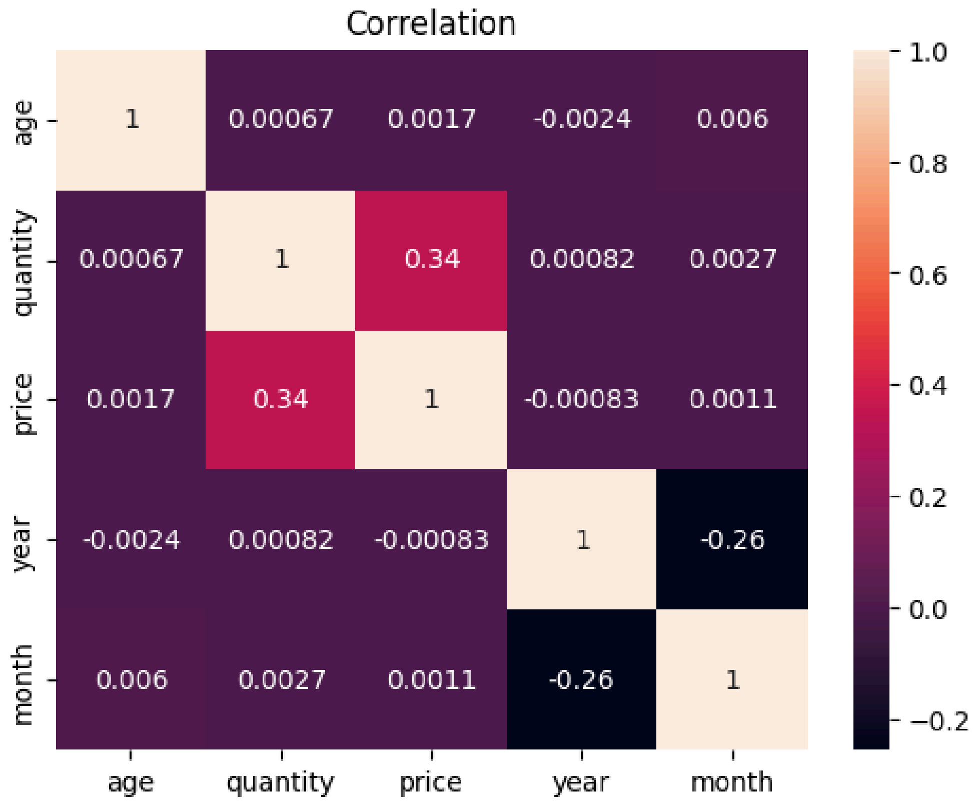
```
sns.barplot(data=months_gender, x='month', y='count', hue='gender',palette=
["#FFB6D9", "#99DBF5"])
plt.xlabel('Month')
plt.ylabel('Order Frequency')
plt.title('Orders in Month')
plt.show()
```



this Graph show the number of Orders in Months

In [46]:

```
sns.heatmap(data=df.select_dtypes(include='number').corr(),annot=True)
plt.title('Correlation')
plt.show()
```



Get the Correlation between the numrical columns